Tampere University

Juha-Matti Mäkitalo

SYSTEM SIMULATION OF FOREST MACHINES

Innovative Hydraulics and Automation
Master's Thesis
October 2020

ABSTRACT

Juha-Matti Mäkitalo: System simulation of forest machines
Master's Thesis
Tampere University
Automation Technology
October 2020

This master's thesis has been done in cooperation with Ponsse Plc. The aim of this thesis is to create a basis for Ponsse's system simulation by mapping the simulation software that meet Ponsse's requirements as well as possible and to test the combination of the selected software by simulating a K121 loader. In system simulation, the aim is to simulate different areas in the same simulation model, which are mechanics (multibody dynamics), hydraulics and control system in this thesis. In addition, forest machine's power train, electrics or environment, for example, could be simulated.

Different software options for system simulation were mapped out through visits and meetings with other companies and software suppliers. Also, Mevea simulation software was introduced in the form of a training course.

The selected simulation software are Mevea (mechanics), Amesim (hydraulics) and Simulink (control system). Simulink was the only simulation software that had been chosen before the beginning of this thesis. The combination of the simulation software mentioned above was able to simulate the K121 loader in real-time with a time step of 1 ms. Mevea is a master software, while Amesim and Simulink are slave software. An FMU principle and shared memory area were applied to the interfaces between the different simulation software, and the simulation principle was co-simulation.

Keywords: system simulation, co-simulation, FMU

TIIVISTELMÄ

Juha-Matti Mäkitalo: Metsäkoneiden systeemisimulointi
Diplomityö
Tampereen yliopisto
Automaatiotekniikka
Lokakuu 2020

Tämä diplomityö on tehty yhteistyössä Ponsse Oyj:n kanssa. Työn tavoitteena on luoda pohjaa Ponssen systeemisimuloinnille kartoittamalla mahdollisimman hyvin Ponssen tarpeisiin soveltuvat simulointiohjelmistot ja testata valittujen ohjelmistojen yhdistelmää simuloimalla K121-kuormainta. Systeemisimuloinnissa pyritään simuloimaan samassa simulointimallissa eri osa-alueita, jotka ovat tässä diplomityössä mekaniikka (monikappaledynamiikka), hydrauliikka ja ohjausjärjestelmä. Lisäksi voitaisiin simuloida esimerkiksi metsäkoneen voimansiirtoa, sähköjä tai ympäristöä.

Eri ohjelmistovaihtoehtoja systeemisimulointia varten kartoitettiin käymällä yritysvierailuilla ja järjestämällä palavereita muiden yritysten ja ohjelmistotoimittajien kanssa. Mevean simulointiohjelmistoon tutustuttiin myös koulutuksen muodossa.

Valitut simulointiohjelmistot ovat Mevea (mekaniikka), Amesim (hydrauliikka) ja Simulink (ohjausjärjestelmä). Simulink oli ainoa simulointiohjelmisto, joka oli päätetty valita ennen diplomityön aloittamista. Edellä mainittujen simulointiohjelmistojen yhdistelmällä pystyttiin simuloida K121-kuormainta reaaliajassa 1 millisekunnin aika-askeleella. Simulointiohjelmistoista Mevea on master-ohjelma, kun taas Amesim ja Simulink ovat slave-ohjelmistoja. Eri simulointiohjelmistojen välisiin rajapintoihin sovellettiin FMU-periaatetta sekä jaettua muistialuetta ja simulointiperiaatteena oli co-simulointi.

Avainsanat: systeemisimulointi, co-simulointi, FMU

PREFACE

I want to thank my supervisors Taneli Heikkilä and Prof. Jouni Mattila for their important advice.

Tampere, 19th October 2020

Juha-Matti Mäkitalo

CONTENTS

LIST OF SYMBOLS AND ABBREVIATIONS

API	Application programming interface
CAD	Computer-aided design
FMI	Functional mock-up interface
FMU	Functional mock-up unit
HIL	Hardware-in-the-loop
HuIL	Human-in-the-loop
MIL	Model-in-the-loop
PIL	Processor-in-the-loop
RUL	Remaining useful life
SIL	Software-in-the-loop
UML	Unified modeling language
XML	Extensible markup language

1. INTRODUCTION

Simulation has been used in product development for several decades. Simulation means imitating a real-world process or system on a computer. Simulation has become an important part of the design of a new product or the improvement of an old one, as it has been proven to save costs and speed up the design phase of the work.

The aim of this thesis is to form a basis for forest machines' system simulation for Ponsse Plc. In other words, it means that different working areas can be simulated in the same simulation model and with different simulation software if needed, for example, hydraulics and control system could be simulated with different simulation software at the same time in the same model. The areas simulated in this thesis are mechanics, hydraulics and control system. When more than one software is used in the simulation, it is important that the interfaces between the software work seamlessly. In the past, different areas have been simulated at Ponsse separately, but not together. This is one reason why it is desired to form the basis for system simulation. In order to form the basis for system simulation, it is necessary to survey the options of different simulation software and choose the suitable combination of them that meet the Ponsse's requirements well. The requirements set by Ponsse include for instance real-time simulation and the ability to simulate the environment surrounding a forest machine.

The structure of this thesis is as follows: first the topic is approached from a theoretical point of view, then the course of the simulation software selection process is presented and what options were reached, and finally the results, further development and conclusions are presented.

2. THEORETICAL BACKGROUND

This chapter approaches the topic of this thesis from a theoretical perspective. The theory consists of model-based design, simulation, different testing methods, interfaces and the advantages and disadvantages of simulation.

2.1 Model-based design

Model-based design is used to help design complex systems. The idea of model-based design is that systems can be modelled on a computer and no physical prototypes are needed early in the design process. The building of useless prototypes can be avoided because model-based design can utilize simulation and automatic code generation at component-level and system-level. Automatic code generation eliminates errors made by human compared to writing code manually. One strength in model-based design is also that existing models can be reused if, for example, the design of a new product includes some of the same features as the previous product [1]. The workflow in model based-design is iterative, so the working principle is tested continuously at different phases. When an error is detected, it is possible to return to the previous phase of the design process. [2]

The workflow of model-based design can be described by a V-model. V-model is sometimes also referred to as V-cycle. V-model is presented in figure 1.

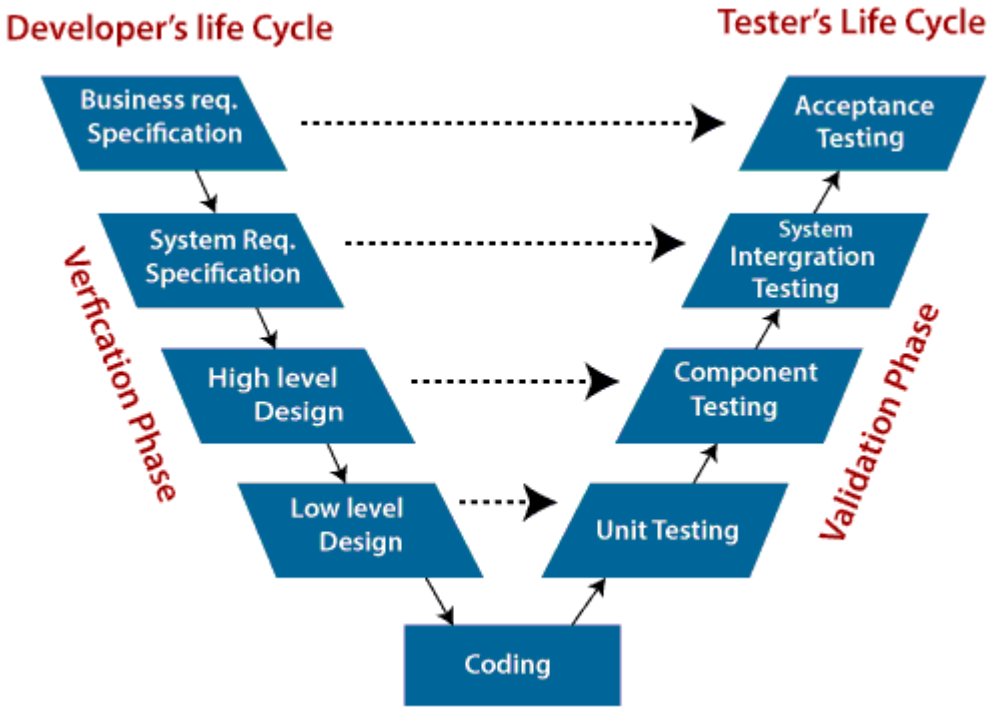

Figure 1. *V-cycle of workflow in model-based design [3].*

V-model, also known as verification and validation model, shows that testing of a system is done in parallel with a certain developing phase (horizontal dashed arrows). V-model is quite similar to waterfall model. The waterfall model has the same phases as the V-model, but unlike the V-model, the testing is performed only at the end. In V-model, verification determines whether the system meets the specific requirements that have been set for it and validation determines whether the software meets the requirements set for it. Thus, the software code is executed only in the validation phase. [3] Model-based design does not always include software, for example if mechanical structure or hydraulic components of a model are modelled without writing code manually.

Verification phase begins with gathering the requirements for the system from a customer, for instance. After that, the next step is to begin analysing the requirements and designing the system based on them. In the high-level design, the architecture of the system is broadly designed, for example modules, their operation in brief and the interfaces between them. The next phase is called low-level design, where, as the name implies, the operation of the system is planned in detail. The last step in V-model's verification phase is coding, in which for instance the operation of the system can be implemented in a suitable programming language if needed. This could be done manually or with simulation software's automatic code generation. [3]

Validation phase begins with unit testing phase, which means that for example modules are tested separately from the rest of the system. In this way it is ensured that the small

different units of the system work as desired. In the component testing phase, the interfaces of the smallest parts of the system are tested to ensure that they work and can transmit information to other units. Unit and component testing are followed by system integration testing, where it is tested that the whole system with its components works as it should. The final step in V-model is acceptance testing. In acceptance testing the system is tested in user atmosphere, which discloses compatibility problems with the user's current systems. [3]

2.2 Simulation

Simulation is imitation of a real-world process or system and it is used to predict how the course of a process could proceed. The aim of simulation is to make the simulation model accurate enough to study a certain phenomenon, as it takes a lot of time and work to make an unnecessarily accurate model due to the adjustment of its numerous parameters. This thesis focuses on modern computational simulation performed on a computer. In order to be able to use the simulation model for how a system could behave, some basis must be set first for the model, which it follows. In computer simulation, such a basis is for example a law of real-world physics. Thus, computer simulation can be thought of as equation-based simulation. [4][5] The passage of the simulation time can also be adjusted to study really fast or really slow processes that might not provide useful information in real-time.

When the time course of the simulation is as fast as the time course of the real world, it is called real-time simulation. In real-time simulation, the computation and data transfer must take place before a certain deadline. Real-time simulation can be divided into three different categories based on deadlines: hard, firm and soft real-time simulation. In hard real-time simulation, the requirement is that all computation and data transfer always take place within the time step without any delays. In firm real-time simulation the computation and other tasks may exceed the time step deadline, but after that no useful information is obtained for the system. Also, in soft real-time simulation the time step deadline can be exceeded, but the usefulness of the results decreases the more the time step deadline is exceeded. [6]

2.2.1 System simulation

This section introduces mechatronic system simulation. System simulation is sometimes also called multi-domain simulation. According to Siemens PLM Software, a system is a group of multi-domain or multi-physics components interacting together [7]. An example of a system's different areas to be simulated is presented in figure 2.

Figure 2. *System simulation with different areas.*

In system simulation, a system may consist of for example control system, electrics, hydraulics, mechanics and environment, because system simulation aims to simulate the whole system and not just one of its areas. System simulation may not be done with just one simulation software. System simulation aims to simulate different areas accurately, so it is possible to combine the best features of different simulation software and utilize them when performing simulations. In other words, this means that if a software has for example high-quality capabilities and comprehensive modelling libraries for hydraulic simulations but not mechanical simulations, then only the hydraulic modelling could be done with this software and the mechanics could be modelled with some other software. [8]

2.2.2 Interfaces, model exchange and co-simulation

Simulation software use interfaces to communicate with each other. Although interfaces make it possible to use the best features of different simulation tools, making interfaces can be laborious, challenging and a combination of many different software increase latency in data transfer. This section introduces two different principles of how interfaces are used between simulation software. The first way to use interfaces is model exchange and its working principle is described in figure 3.

Functional mock-up unit (FMU), shown in figure 3, is an external simulation model ZIP file that includes extensible markup language (XML) file and application programming interface (API) that consists of standardized functions in C language. The XML file describes the behavior of a model, such as variables and parameters. The FMU can be connected to other simulation software via functional mock-up interface (FMI). [9]

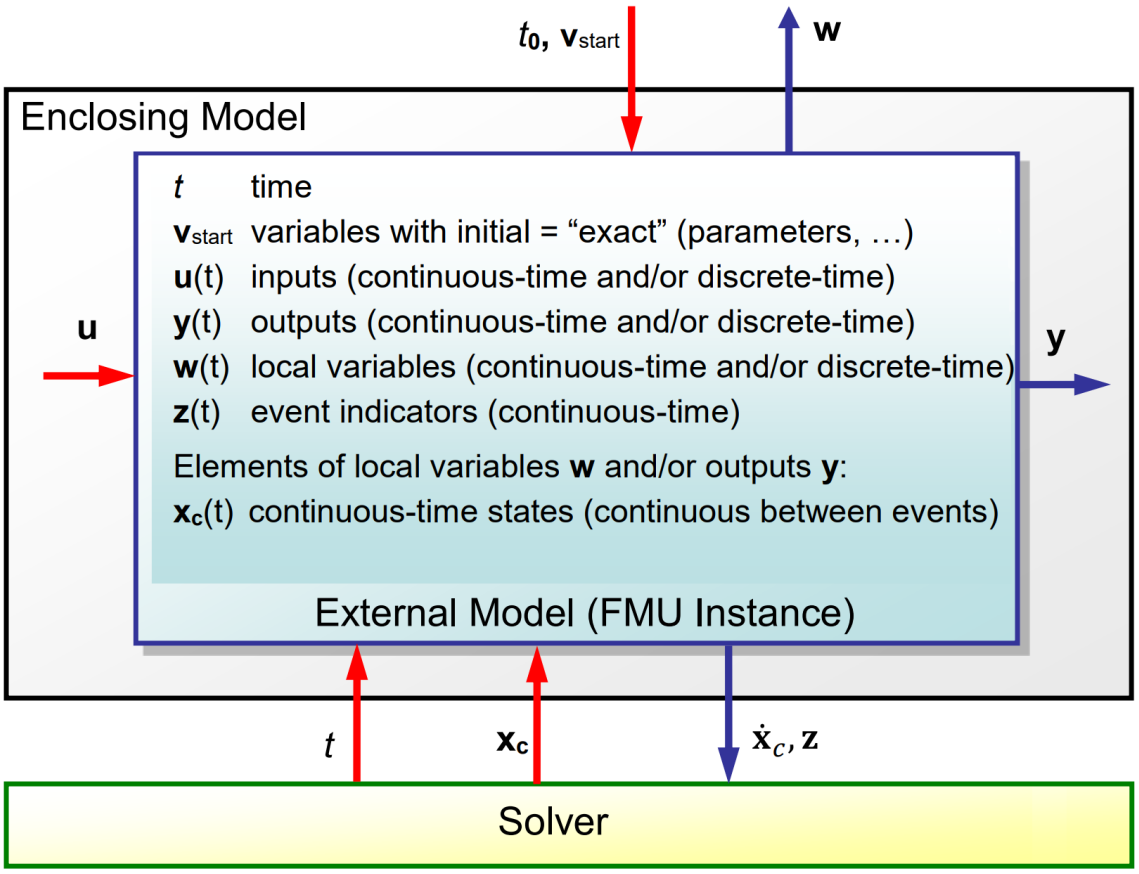

Figure 3. *Model exchange working principle [9].*

Model exchange's key feature is that the external model is provided without its own solver as an FMU. In model exchange the input (u) comes from enclosing model and then the FMU interacts with the external solver to perform integrations and calculate the output (y). This sequence is executed at each time step. The whole system may have some local variables (w) that can be used by all models. In the beginning of the simulation (t_0), the possible initial parameters (for example v_{start}) are entered into the system on the first time step. Model exchange is one option to use functional mock-up units, but it was not utilized in this thesis. The second way to use interfaces is co-simulation and its working principle is described in figure 4.

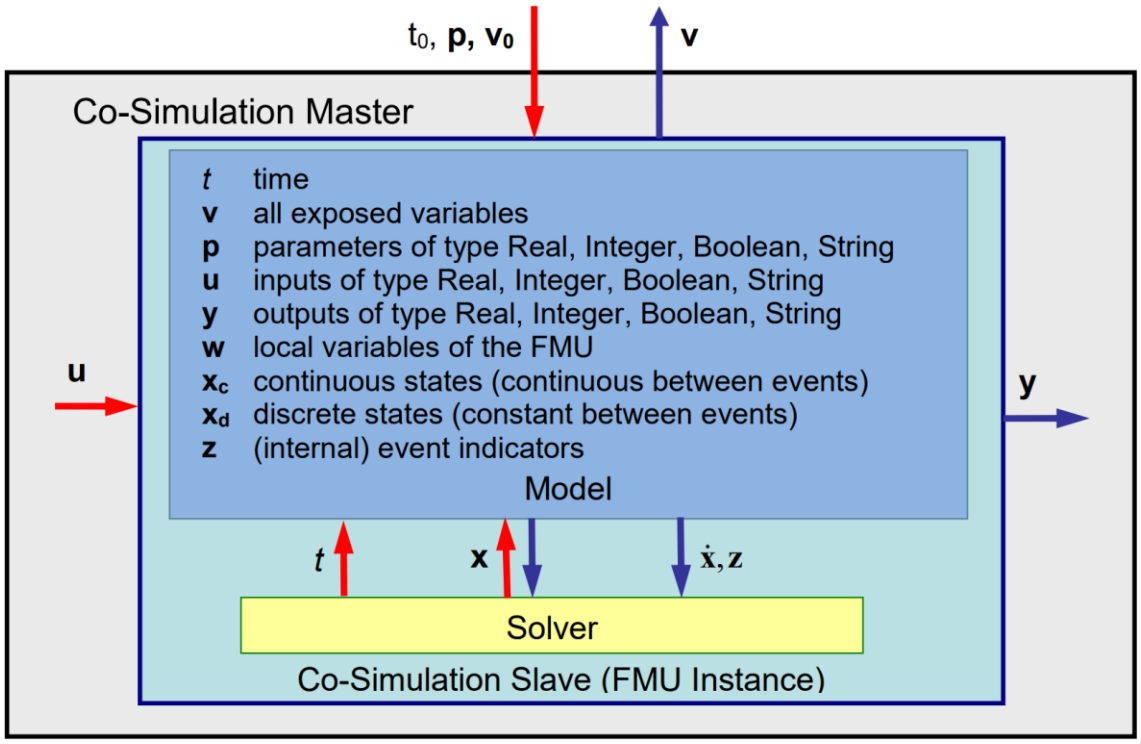

Figure 4. *Co-simulation working principle [9].*

The clearest difference between model exchange and co-simulation is that in co-simulation the FMU includes its own solver as it can be seen in figure 4. Now the integrations are performed inside the FMU instead of having to use an external solver. The software to which the FMU is connected to is called master and the FMU model is called slave. This means that all data flow between different functional mock-up units takes place via the master software. Also in figure 3, the “Enclosing Model” is the master software and the “External Model” is the slave software. In this thesis, the simulations were performed according to the co-simulation principle, which is described in more detail in section 4.4. The co-simulation technique has two different ways of using FMU. The first one, co-simulation with generated code, is presented in figure 5.

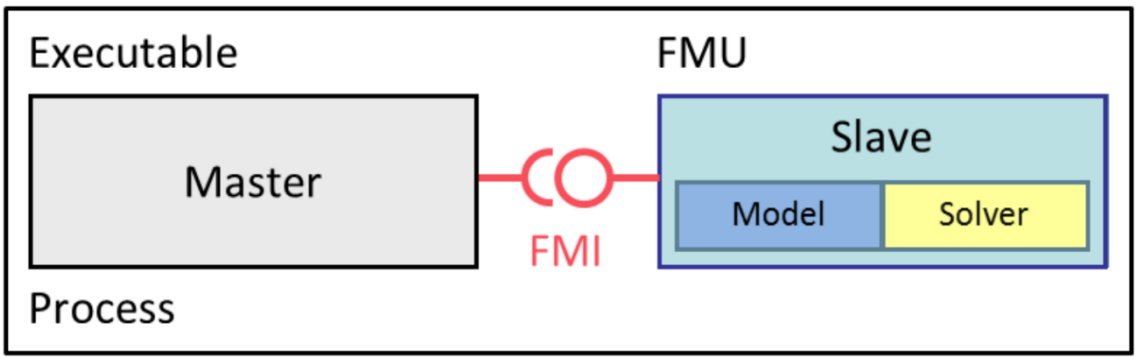

Figure 5. *Co-simulation architecture with generated code [9].*

In the co-simulation architecture in figure 5 the FMU is exported as generated code to master software. In this case, the slave software does not need to be running during the simulation. The second co-simulation technique, co-simulation with tool coupling, is presented in figure 6.

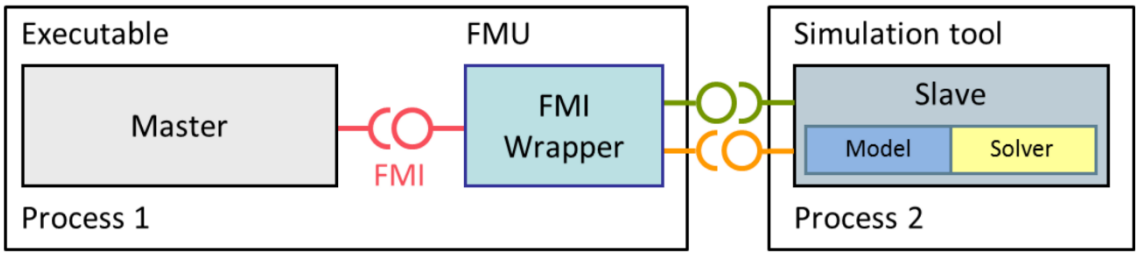

Figure 6. *Co-simulation architecture with tool coupling [9].*

The main difference between generated code and tool coupling is that in tool coupling the slave simulation tool must be running. Also, there is an “FMI wrapper” that includes for example files in C language that can be compiled to form the FMI. The interface between FMU and simulation tool is application programming interface (API) that is created by simulation tool and connected to FMI. The application programming interface could be for example shared memory area that can be accessed by both master and slave.

If the computation of the simulation model becomes too heavy for one PC, the simulation models can be distributed to several PCs. The architecture of distributed co-simulation is presented in figure 7.

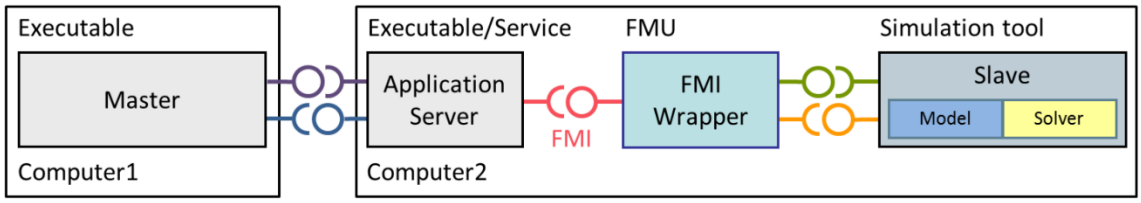

Figure 7. *Co-simulation architecture with two different computers [9].*

In figure 7 there are two different PCs, Computer1 and Computer2. Computer1 takes care of master model’s computation and the communication layer between Computer1 and Computer2. Instead, Computer2 handles the computation in slave model, FMI wrapping and maintains the application server. The communication layer between Computer1 and Computer2 is a different matter from the interfaces of the FMI standard. The communication layer needs parameters, for example identification of the remote computer, port number and user account, and they are set via graphical user interface in the master PC. [9]

Although there is only one slave in figures 5-7, it is possible that there would be several of them. For instance, there are two slave models in the simulation model of this thesis. The model is presented later in chapter 4.

2.2.3 MIL, HuIL, SIL, PIL and HIL testing

This section describes different testing methods for simulation models. The testing methods belong to the verification part of the V-cycle presented in figure 1. The abbreviations used in the title of this section are explained below:

- Model-in-the-loop (MIL)
- Human-in-the-loop (HuIL)
- Software-in-the-loop (SIL)
- Processor-in-the-loop (PIL)
- Hardware-in-the-loop (HIL)

When making a simulation model, the model must be tested iteratively between different testing phases to find errors and other unwanted properties. Then it also can be seen what is already working properly in the simulation model and what no longer need to be changed as the modelling process continues. The first testing method presented is model-in-the-loop, in which a simulation model of the system is built. In this phase, the system does not yet include any physical components. [10]

When a simulation model is tested, it must be controlled. During the simulation, the control can be handled with signal builders, for example, or the user of the model can control the system himself. If the user controls the system himself, it is considered human-in-the-loop testing method. The HuIL testing method is used throughout the simulation model development process unless the system is autonomous and thus does not require human guidance to operate.

Another testing method is called software-in-the-loop. At this phase, for example, the operation of the controller can be included in the simulation model as generated code. The code runs on the same PC as the simulation model. [11][12] Also, the inclusion of another simulation software simulation tool in the system in code format (FMU) can be considered as utilizing the SIL testing method. At SIL testing, it is not necessary to achieve real-time simulation, because the model is still running on normal PC where the model has been built, instead of a powerful real-time computer.

Processor-in-the-loop method is quite similar to the SIL method, but now the software tested in the SIL phase is tested with a separate embedded processor. Thus, it is important that the interface between the embedded processor and the earlier simulation environment can be tested and made operational. In the PIL phase, the simulation also does not need to be capable of running in real-time. [11][12]

Compared to the methods presented above, hardware-in-the-loop is closest to the finished product or system. In HIL testing, the final and powerful hardware of the simulation environment is added to the system. At this point, the goal is to achieve real-time simulation with the actual hardware. [10][11]

This thesis focuses on MIL, HuIL and SIL testing. In the further development of the simulation model of this thesis, PIL and HIL testing methods would become essential.

2.2.4 Digital twin and predictive maintenance

There are many definitions for the word digital twin because it is not easy to define unequivocally. One example of digital twin is presented in figure 8.

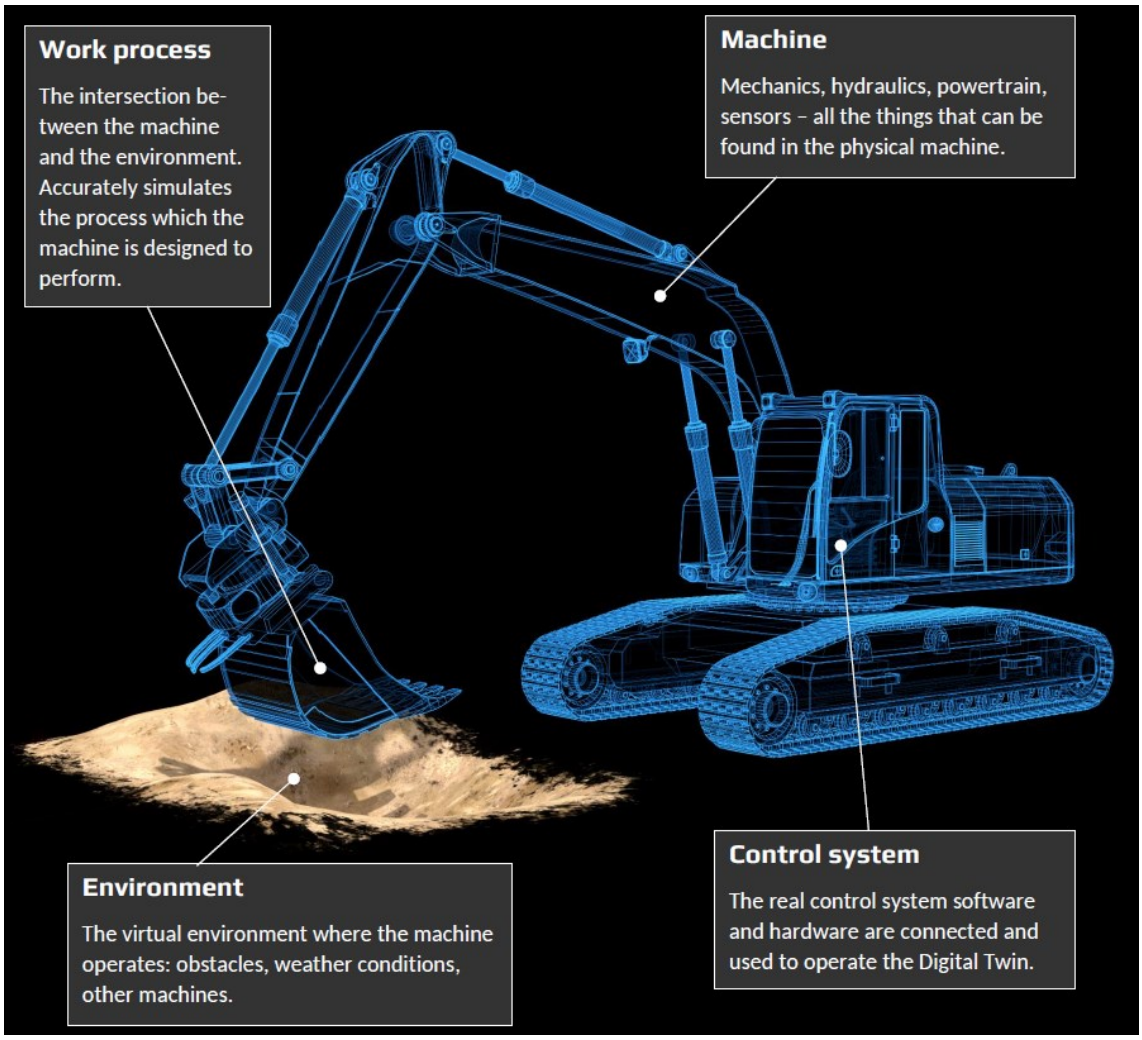

Figure 8. *Physics-based digital twin [13].*

A digital representation of a product or a system is called a digital twin that is formed on the basis of a physical twin using simulation models with real-life data [14]. According to Mevea Ltd, a digital twin is a virtual model of an existing product that includes mechanics, hydraulics, powertrain, sensors and control system. The virtual model can be simulated in virtual world, so the digital twin also includes modelled terrain and machine's interaction with it. The digital twin can be developed during the whole life cycle of the machine. [13]

By developing digital twin well and accurately, the simulation of physics-based models can enable predictive maintenance. Predictive maintenance means that with an exact simulation model of a product, the simulation model can foretell for example if some component of the system is going to break. In other words, with predictive maintenance it is possible to obtain remaining useful life (RUL) of a system or a component. [15]

2.2.5 Advantages and disadvantages in simulation

There are numerous advantages and disadvantages in simulation, which are discussed in this section. One advantage, for example, is that simulating a system is very safe. When the simulation model is run on a computer screen instead of a real-life physical prototype, for instance personal injuries are avoided in case the system or component breaks down or collides with something. This facilitates, for example, the early development of a new product before the prototype is built, so that the impact of varying and different sized components on system performance can be tested more safely, cheaper and faster than with a built prototype. Simulation is cheaper because there is no need to buy new components to be tested and simulation is faster because it does not take so much time to test new components in the system; only a few parameters are changed, for example the diameter of a cylinder, instead of installing a physical component in place in the real-life system. In addition, the use of various training simulators is smart, for instance in the training of forest machine drivers, which causes savings in fuel costs.

There are also some disadvantages in simulation, one of which is that making a simulation model can take a really long time to be accurate enough. In other words, accurate enough means that the simulation model aims to obtain results that would be useful, for instance, in product development. Some things and phenomena can be difficult to model in a simulation model, such as friction. It is also challenging to make the simulation model accurate in every single area, so it is usually done by modelling some desired area accurately and in other areas the model is a little less accurate. In company, an employee with good knowledge is also required to interpret and process the simulation models, so not everyone can benefit from the simulation model. This leads to the conclusion that in order for other than the creator of the simulation model to be able to utilize the simulation model, a functional version control is needed. In version control it is important that employees know which version of the simulation model they can modify themselves. Poor version control can result in someone other than the creator of the simulation model modifying the original simulation model and it no longer works as desired.

3. SURVEY OF SYSTEM SIMULATION SOFTWARE

In this chapter the company, the selection process and researched simulation tools are presented. Also, the final combination of the chosen simulation tools is explained.

3.1 Company background

Ponsse Plc was founded in 1970 and nowadays it is one of the world's leading manufacturers of cut-to-length forest machines. The cut-to-length method is an environmentally friendly logging method, where the harvester fells, delimbs and measures the trees, and then cuts them into desired length ordered by the mill [16]. Ponsse operates in harvesting markets in over 40 different countries and its main products are harvesters and forwarders. All Ponsse forest machines are manufactured in Vieremä, Finland. Nearly 80 percent of Ponsse's net sales come from exports. Ponsse has many subsidiary companies around the world, for example in Sweden, Norway, France, UK, Ireland, Russia, USA, Brazil, China and Uruguay. [17]

3.1.1 K121 loader crane

The loader that is simulated in this thesis is Ponsse K121. It was released in 2018 and it is available for two largest Ponsse forwarders: Ponsse Elephant and Ponsse ElephantKing. Ponsse K121 loader is presented in figure 9.

Figure 9. *Ponsse ElephantKing forwarder with K121 loader.*

The forwarder in figure 9 belongs to one of Ponsse's customer and it has much more sensors than forwarders usually have. There have been installed extra pressure sensors and strain gauges all over the forwarder, so it is possible to get more data with better accuracy for simulations.

The K121 loader is the biggest loader that Ponsse has manufactured. With K121 loader it is much more convenient to work in steep environments because the tilt stand has larger tilt angle than the previous loader models. The technical information of K121 loader is presented below:

- Lifting torque: 195 kNm.
- Slewing torque: 48 kNm.
- Reach: 8 m/10 m (S/M models).
- Tilting base tilt angle: -12°…+24°.
- Large grapple surface area: 0,50 m^2. [18]

3.1.2 Simulation background

Before this thesis, Ponsse has done simulations in a few areas separately. For example, there have been hydraulics and control system simulations. Also, there are simulators for training purposes.

Hydraulics simulations have been performed by using Simcenter Amesim that is developed by Siemens PLM Software. For hydraulic simulations, models made by component suppliers have been obtained for some components, such as valves.

Control system logic has been tested and simulated with MathWorks' MATLAB and Simulink. This simulation software had been selected to be included in the final combination of system simulation tools before this thesis, but the other simulation software had not yet been chosen.

Training simulators have been developed in order to train forest machine drivers. Training simulators do not include hydraulics, so the user of the simulator only controls for example the movement of the boom, not hydraulic flow or valve openings. Training simulator developed by Creanex is presented in figure 10.

Figure 10. *Ponsse training simulator.*

The training simulator includes joysticks and user interface that are the same as on Ponsse's real-life forest machines. The training simulator in figure 10 only looks visually realistic, so it cannot be used to determine for example the magnitudes of forces or loads on the crane or pressures in cylinders

3.2 Selection process

During the selection process there were many meetings with software suppliers. There were also plenty of other companies that were visited, for example Sandvik, Kalmar, Hiab and Creanex. The companies gave information on which simulation tools they had been using and what had been the advantages and disadvantages in them.

3.3 Researched simulation tools

The simulation tools that were researched were Ansys Motion and Ansys Twin Builder, Adams, Simscape Multibody, Simcenter Amesim and Mevea. Each of the mentioned software had been in use at some of the companies that were visited during the thesis.

Ansys Motion and Ansys Twin Builder were introduced by EDR Medeso. Ansys Motion is a multibody dynamics simulation software for both rigid and flexible bodies and Ansys Twin Builder is a system simulation software that can be used to perform multi-domain simulations. Twin Builder supports many common languages and exchange formats, for instance C/C++ and FMI. [19][20] According to EDR Medeso, Motion has ready-made tire and soil models, but is not capable of real-time simulation with them. EDR Medeso also stated that Twin Builder has different libraries for simulations that are made in Modelica language that is an open source language made for modelling components for system simulation.

Adams (Automatic Dynamic Analysis of Mechanical Systems) was presented briefly by Patria in an online meeting. Adams is developed by MSC Software and it is also designed for multibody dynamics simulation. According to MSC Software, Adams is the most popular simulation software in the world. Adams includes many different solutions, for example Adams Controls, Adams Durability, Adams Mechatronics, Adams Vibration, Adams Flex, Adams ViewFlex, Adams Car and Adams Driveline. With this many solutions, it is very likely that it is possible to model exactly the area you want. Adams is capable of real-time simulation and flexible bodies. A control system for simulation model can also be modelled using it. [21]

Simscape Multibody (formerly SimMechanics) is, as the name implies, a multibody dynamics tool made by MathWorks. The features and abilities of the software were introduced by MathWorks' employees. According to them, it is not necessary to create separate collision graphics, because the bodies' own graphics are used in contacts. This is computationally quite cumbersome, so achieving real-time simulation could be challenging. Simscape Multibody has an import tool for CAD models that gathers masses, iner-

tias, joint locations and 3D graphics for the mechanical simulation model. The CAD import tool speeds up the work considerably, but it can sometimes contain errors as well if the CAD geometry is complex. Parameters can be set for Simscape Multibody via MATLAB and it works seamlessly with Simulink. Simscape also has a library for simulating hydraulics, Simscape Fluids, that includes ready-made models for hydraulic component, such as valves and orifices. [22]

Simcenter Amesim is developed by Siemens PLM Software and it is a system simulation platform that is specified for mechatronic simulations. Simcenter Amesim includes comprehensive libraries for different applications in the industry, for example electrical, fluid and mechanical fields. Amesim can be connected to various other software, for example CAD software and it can also utilize FMI connections. [23]

Mevea simulation software is developed by Mevea Ltd and it is designed for mechanics modelling and simulation. With Mevea, it is also possible to simulate hydraulics and connect the real-life hardware control system to the simulation. Mevea uses two different programs to simulate: Mevea Modeller and Mevea Solver. Mevea Modeller is used to build simulation models and Mevea Solver is used to simulate and visualize them. [24]

3.4 Selected software

As mentioned before, Simulink was the only software that had been chosen before this thesis to be included in the final combination of simulation tools. The other chosen simulation software were Mevea and Amesim. The final combination is Simulink for control system simulations, Mevea for mechanics simulations and Amesim for hydraulics simulations. For this combination of simulation tools, it had to be made sure before the selection decision that they were capable of real-time simulation together, which was confirmed by Mevea Ltd.

Mevea was chosen for mechanical modelling because it was reported to be capable of simulating contacts between tires and soil in real-time. While visiting other companies, there were some impressive simulators modelled with Mevea, which could be tested. There are also simulators modelled with Mevea by other forest machine manufacturers, which have been successful in simulating for example sawing, felling and delimbing trees. Mevea also held a five-day training course in how to use the software, so there was a lower threshold to start using it. Mevea could also be used to model hydraulics, but it is simplified compared to Amesim's hydraulic modelling. In Mevea the medium is ideal and does not take into account for example its warming.

The choice of Amesim was facilitated by the fact that Ponsse employees have previous experience of using the software, which greatly contributes to the deployment process of the software. According to them, Amesim is one of the best software for hydraulic simulations, because it can be used to model hydraulics very accurately when needed and pre-made models from component suppliers speed up the modelling work significantly. The user interface of Amesim is also very intuitive and clear and the tutorials and documentations are comprehensive.

Ansys Twin Builder could have been very qualified for Ponsse's system simulation needs, but the crucial feature was that real-time simulation could not be achieved with Twin Builder. Ansys Motion and Twin Builder licenses were also quite expensive compared to Amesim and Mevea licenses.

The combination of three different simulation software has both advantages and disadvantages. Three different software and their licenses allow three different employees to work simultaneously, unlike if there were only one software and one license. On the other hand, the total cost of three different licenses may be higher than the price of a broader license for one simulation software.

4. SIMULATION MODEL WITH THE CHOSEN SOFTWARE

In this chapter the simulation models that are made with the chosen software are presented. The main focus was on modelling the mechanical simulation model, because Mevea was the only unfamiliar software to Ponsse.

4.1 Simulation model of mechanics

The simulation model of mechanics was modelled with Mevea software. Mevea user interface is presented in figure 11.

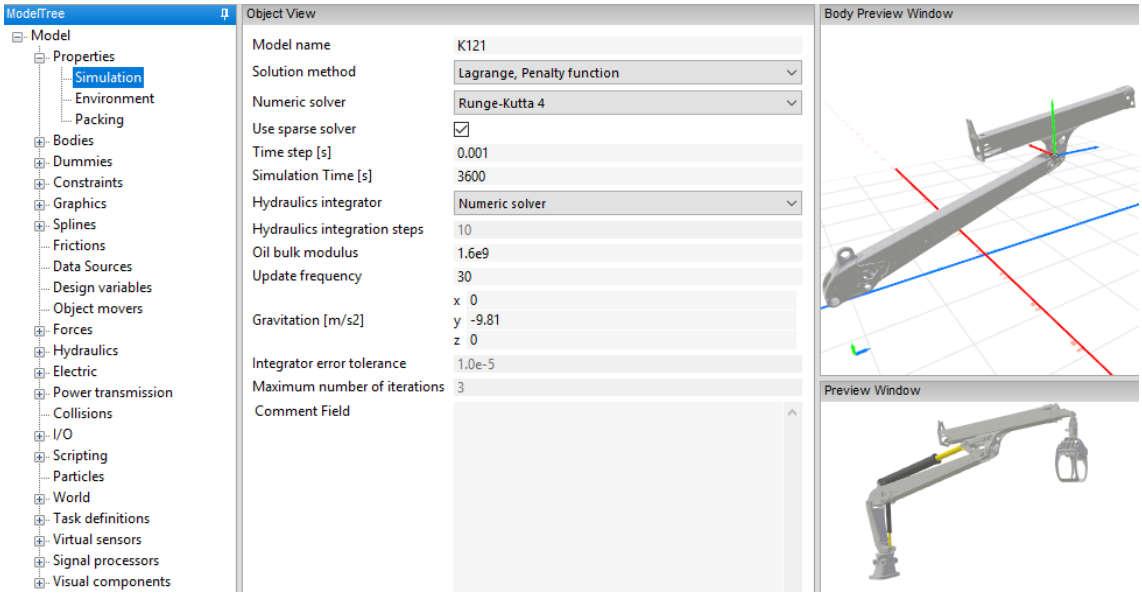

Figure 11. *Mevea user interface.*

Mevea user interface consists of model tree, object view, body preview window and preview window. In model tree a certain folder, tab or component can be selected and then in object view different parameters can be set. In figure 11 it can be seen that from “Properties” folder “Simulation” tab has been selected and in the object view the simulation parameters can be examined and modified. The simulation parameters seen in figure 11 are used in this simulation model. A certain component, body, constraint, force etc. can be seen in body preview window. The body preview window in figure 11 shows a constraint that depicts a revolute joint between lift boom and tilt boom. Constraints are presented later in this chapter. In the preview window the full visual appearance of the model is displayed.

The modelling of mechanics began with building the 3D CAD-model. The CAD-model presented in figure 12 had been modelled with Dassault Systèmes' SolidWorks. To achieve real-time simulation, the CAD-model had to be simplified, which means that unnecessary components and parts were removed from the model. Unnecessary components refer to screws, pipes, hoses and other little parts compared to the main parts of the boom that don't act as load-bearing structures. Hydraulics cylinders are also excluded at this point.

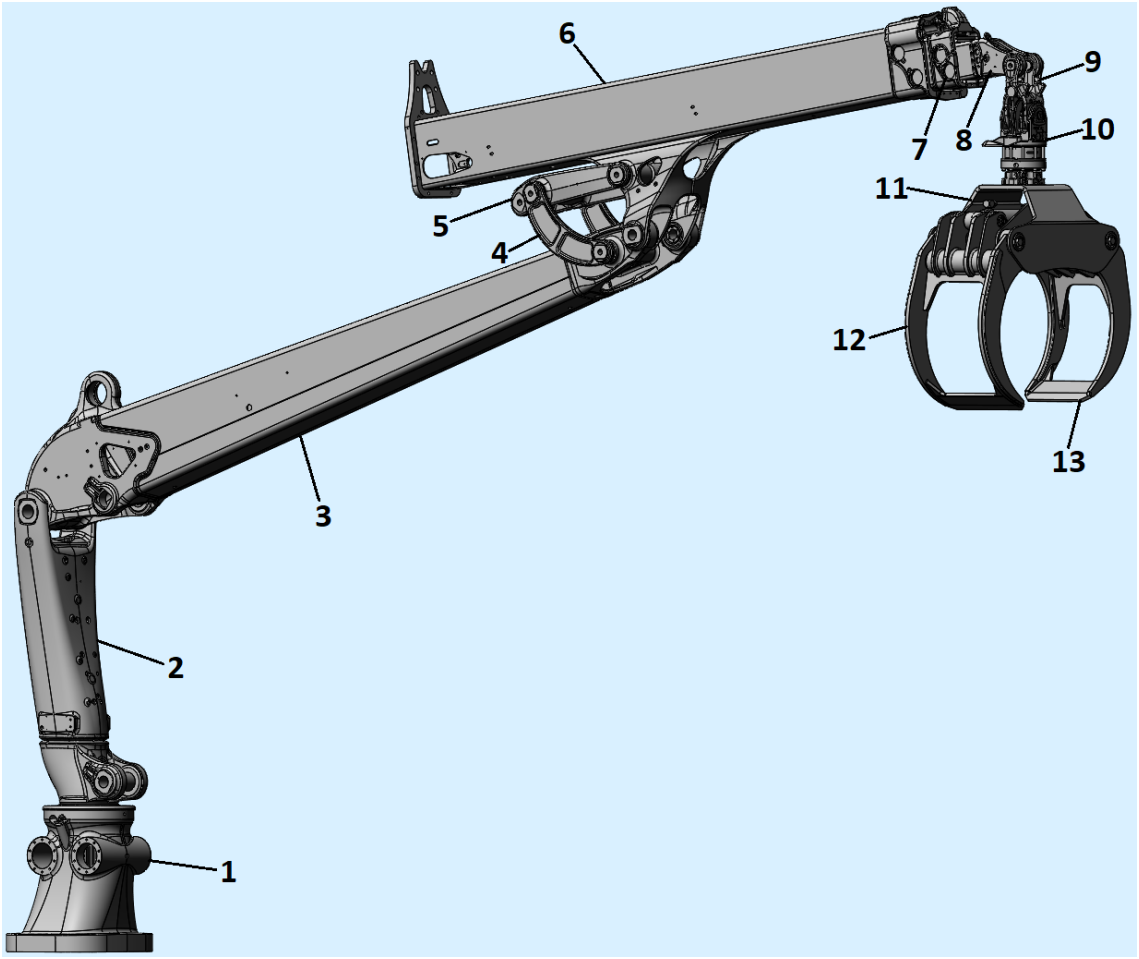

Figure 12. *CAD drawing of K121 loader.*

After the 3D CAD-model had been built, the modelling continued by making bodies in Mevea software. There is a total of 14 bodies from ground toward the tip of the boom, for example the first 4 bodies are ground, base, pillar and lift boom. Ground is a default body and it is invisible in this model and therefore it is not numbered. The bodies of the system are presented in table 1 and the base-body is presented in figure 13.

Table 1. *Body numbers and names.*

Number	Body name
1	Base
2	Pillar
3	Lift boom
4	Reaction rod 1
5	Reaction rod 2
6	Tilt boom
7	Extension 1
8	Extension 2
9	Link
10	Rotator
11	Grapple frame
12	Grapple right
13	Grapple left

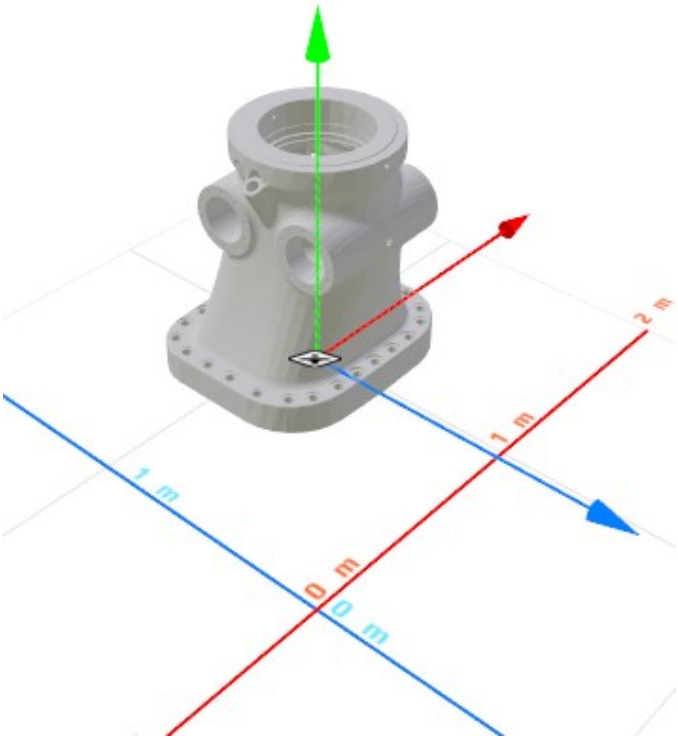

Figure 13. *Base-body preview in Mevea software.*

The graphics of the bodies are imported in stereolithography (.stl) file format in this model. Bodies in STL file format are not as precise as the original CAD-models but STL file format makes the visualization graphics lighter, which is necessary for real-time simulation. Mevea supports also 3D studio (.3ds), OpenSceneGraph (.osgt and .osgb) and Wavefront (.obj) file formats [25]. The red, green and blue arrows represent the local coordinate system (x, y, z) of the body. The imported graphic file does not include any mass properties, only the visual appearance. Therefore, there are several properties that must be set for bodies which are presented in figure 14.

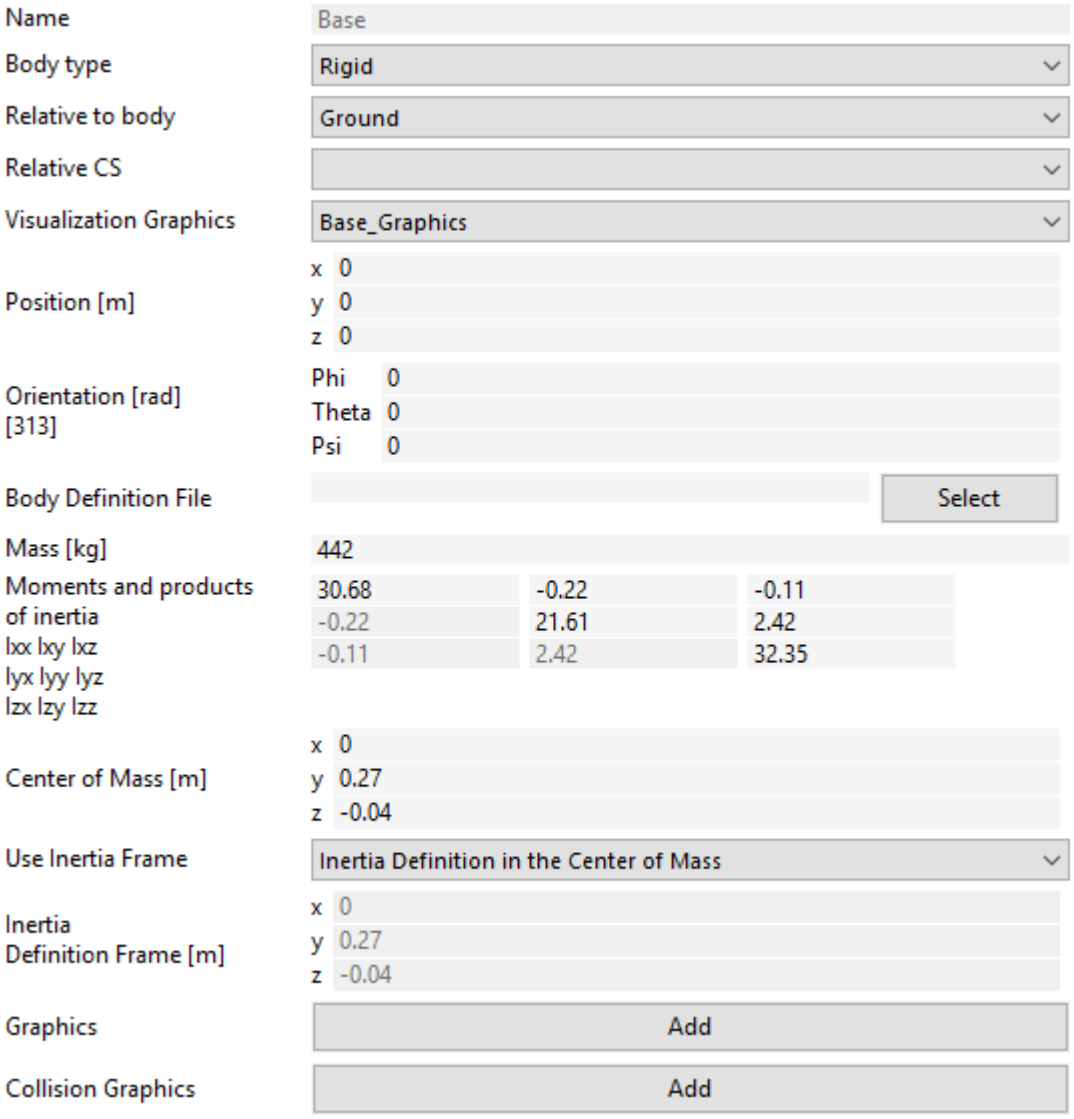

Figure 14. *Body properties in Mevea.*

All the bodies of the system are rigid for simplicity and relative to previous bodies as in figure 14. For example, pillar is relative to base and lift boom is relative to pillar, which forms the kinematic chain. Every body has mass, moments and products of inertia and center mass, which are gathered from 3D CAD-model built in SolidWorks. Unfortunately,

Mevea does not yet have a CAD import tool for SolidWorks that would automatically handle the import of a CAD model into a mechanical model. With the help of import tool of the CAD model, it would not be necessary to manually set for example masses, centers of masses or joint locations.

In real life, there are different joints between boom parts. In Mevea software there are constraints that represent joints. For example, there are revolute and translational constraints in the model. A revolute constraint is presented in figure 15.

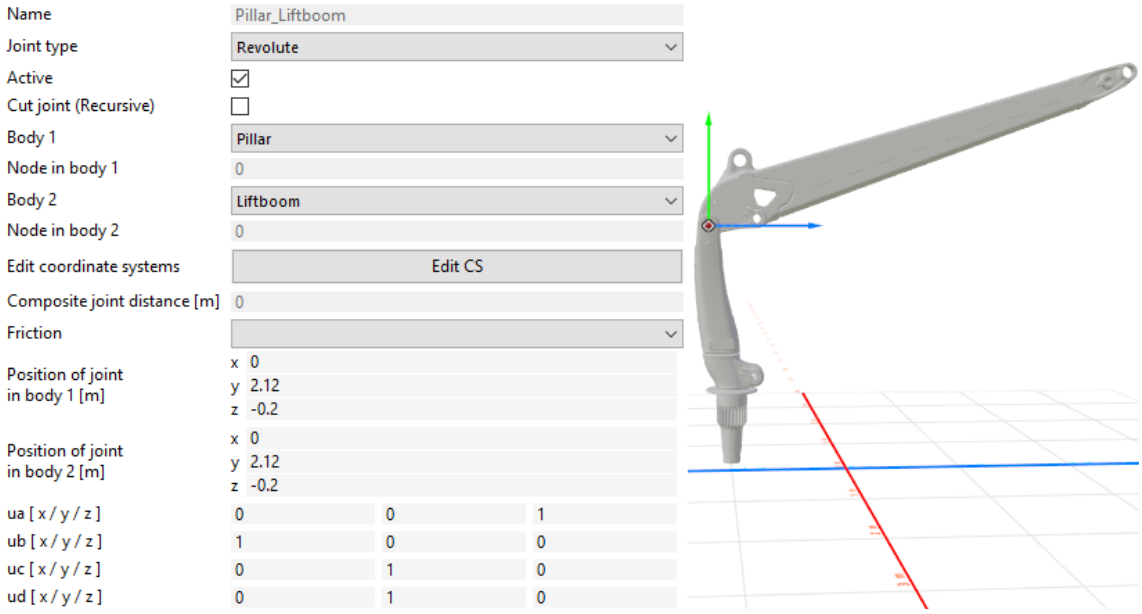

Figure 15. *Revolute constraint between pillar and lift boom.*

In figure 15 it can be seen that there are some properties that have to be defined when making a constraint. For example, joint type (revolute), bodies that are connected to the joint (pillar and lift boom) and joint position (x, y, z). Also, there are unit vectors (ua ub uc ud) that determine the direction of the joint revolution. In order to prevent mistakes, the position of the joint can be visually checked from the preview that can be seen on the right side of figure 15. Most of the joints are revolute joints but there are also two translational joints: between tilt boom and extension 1 and between extension 1 and extension 2 (see figure 12 and table 1). Translational joints allow movement only along a certain vector. In this model, it means that extension bodies can be moved only telescopically according to the tilt boom.

After modelling bodies and constraints, the next step was to model cylinders and cylinder forces. Lift cylinder is presented in figure 16.

Figure 16. *Lift cylinder in Mevea.*

Cylinders' graphics are imported in 3DS file format in this model. These graphics for the cylinder (black) and piston rod (yellow) in figure 16 are from Mevea's tutorial resources. Cylinders and piston rods are modelled as dummies. According to Mevea Ltd, dummies are not similar to bodies: "Dummies are parts which are not connected to the kinematic chain, but affect the mass and inertia properties of the attachment body." [25] This means that for example lift cylinder's mass and inertia properties are considered in lift boom body and lift piston rod's mass and inertia properties are considered in pillar body. It is highly advisable to model cylinders as dummies, because it reduces the number of bodies and constraints. This makes the modelling much simpler and makes the software's computation lighter, which contributes to real-time simulation. Dummies have different types, such as body-to-body force, constraint, static and tyre. In this model the type of cylinder dummies is body-to-body force.

Cylinder forces are modelled as body-to-body forces in this model, which means that the force acts between two certain points in different bodies. For example, lift cylinder's force in figure 16 is attached to act between pillar and lift boom, where the cylinder dummy is located.

Moving the boom requires more than just cylinders and the forces they produce. This leads to the next phase in which the modelling of motors and rotational torques is explained. The modelled torques are produced by motors between certain bodies. In this model there are two rotational torques that are used to control the boom: slewing torque between base and pillar and rotator torque between rotator and grapple frame. In addition, there are several rotational torques that act as joint frictions. Between extension 2 and link bodies and between link and rotator bodies there are also minimum and maximum angles that restrict the rotation of the joints. Other joints' limitations are taken into account in the maximum and minimum lengths of cylinder strokes. Joint friction and motor parameters are presented in figure 17.

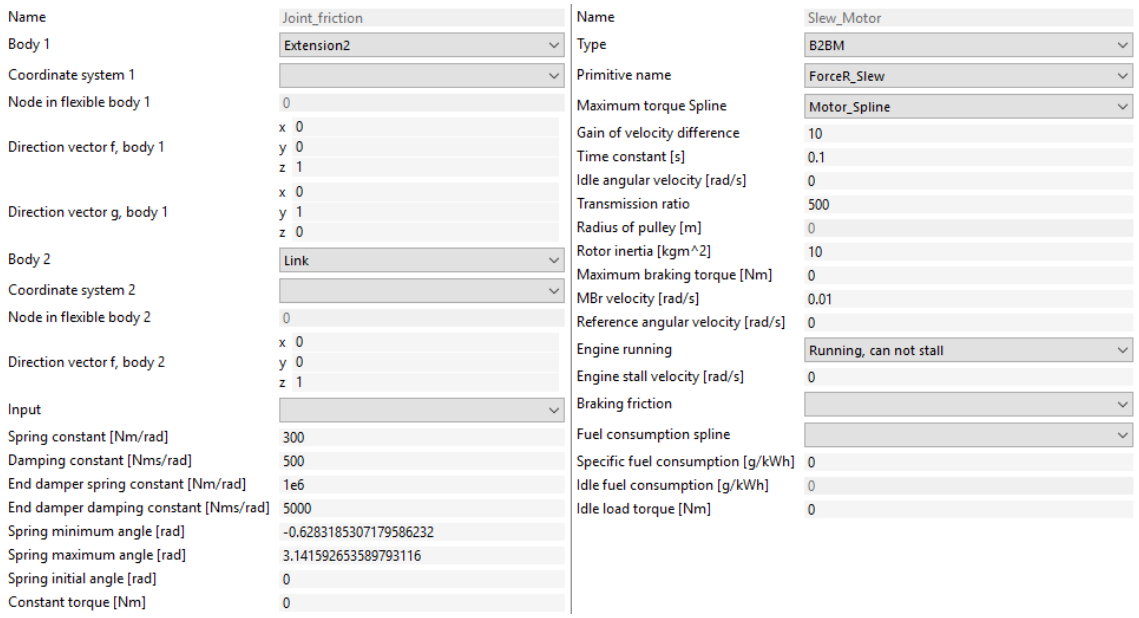

Figure 17. *Rotational torque (joint friction) and motor parameters.*

There are plenty of parameters that can be set for rotational torques and motors. In this model, there are some parameters that were not considered at all. For example, motor's fuel consumption was left out in this thesis. The parameters in figure 17 are not the same as in real life system, because for simplification the parameter values were set in a way that the functionalities of the loader crane could be tested and the boom movements were reasonable. Motors require a torque spline that represents maximum torque as a function of angular velocity. The "Motor_Spline" in figure 17 is not based on real-life slewing motor values, because the boom rotation is modelled with a motor, although it is actually implemented with rotation cylinders.

In order to test and simulate the boom, there must be inputs that can be used to control the boom. Input parameters are presented in figure 18.

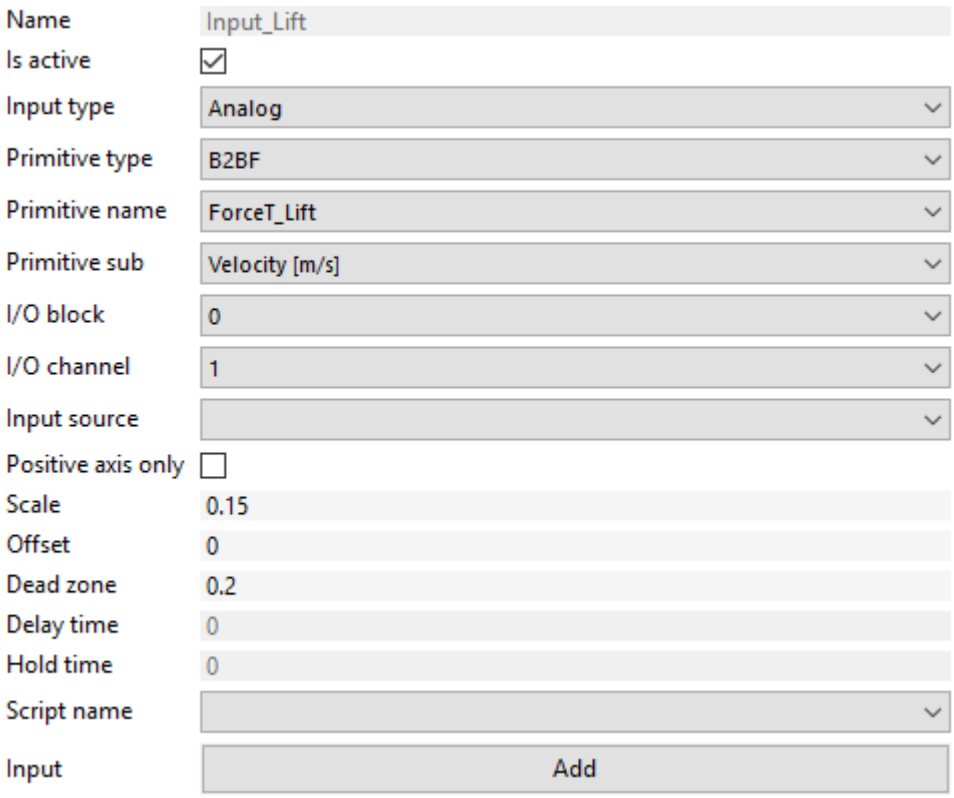

Figure 18. *Input parameters.*

The input parameters of translational forces (lift, tilt, extension, grapple) were all quite similar to each other at this phase of modelling. “Primitive sub” in figure 18 means that the type of the input control signal is velocity. All the inputs had to have different I/O block number and I/O channel number combinations. For example, “Input_Lift” has I/O block number of 0 and I/O channel number of 1, so any other input shouldn’t have the same combination. Scales, offsets and dead zones could be adjusted so that the testing phase of mechanics simulation model went smoothly and controlling of the boom did not cause too high velocities and thus oscillation. After modelling the inputs, the boom was tested and simulated iteratively to obtain reasonable parameter values and to be able to proceed further in the modelling process.

4.2 Simulation model of hydraulics

The simulation model of hydraulics was modelled with Simcenter Amesim. Hydraulic simulation model is quite simplified compared to the real-life hydraulic system, because really accurate modelling of hydraulics was not relevant on this thesis. There are 5 cylinders and 2 motors in the hydraulic model. Cylinders are lift cylinder, tilt cylinder, two extension cylinders and grapple cylinder. For simplicity, the extension cylinder is divided into two separate cylinders, each of which produces force to its own extension body. In

other words, extension cylinder 1 moves extension 1 body and extension cylinder 2 moves extension 2 body. A cylinder model is presented in figure 19.

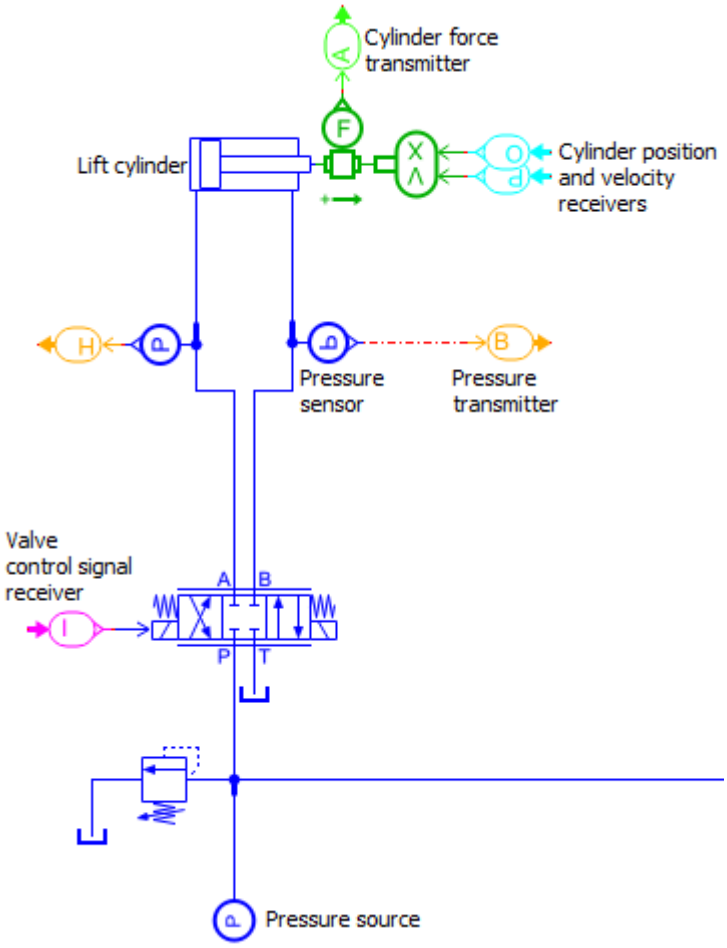

Figure 19. *Cylinder model in Amesim.*

The hydraulic circuit is powered by a simple pressure source that connects to all valves of cylinders and motors. The relevant parameters set for the cylinders were piston diameter, rod diameter, length of stroke, pressures at both cylinder ports, dead volumes at both cylinder ports, viscous friction coefficient, spring rate at endstops and damping coefficient on endstops.

A motor model is presented in figure 20. Transmitters and receivers seen in figure 19 and figure 20 are connected to FMI that is presented later in section 4.4.2.

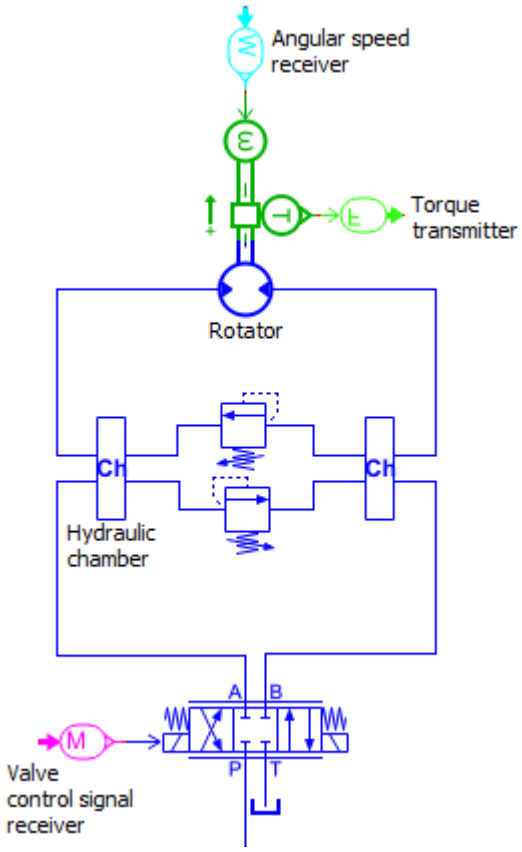

Figure 20. *Motor model in Amesim.*

There are two motor models in the hydraulic circuit: slewing motor and rotator. Hydraulic chambers are added in order to reduce pressure peaks and oscillation.

4.3 Simulation model of control system

The simulation model of control system was modelled with Simulink. The model is very simple, because the main purpose of control system model was to generate signals to control valves in Amesim. The control system model is presented in figure 21.

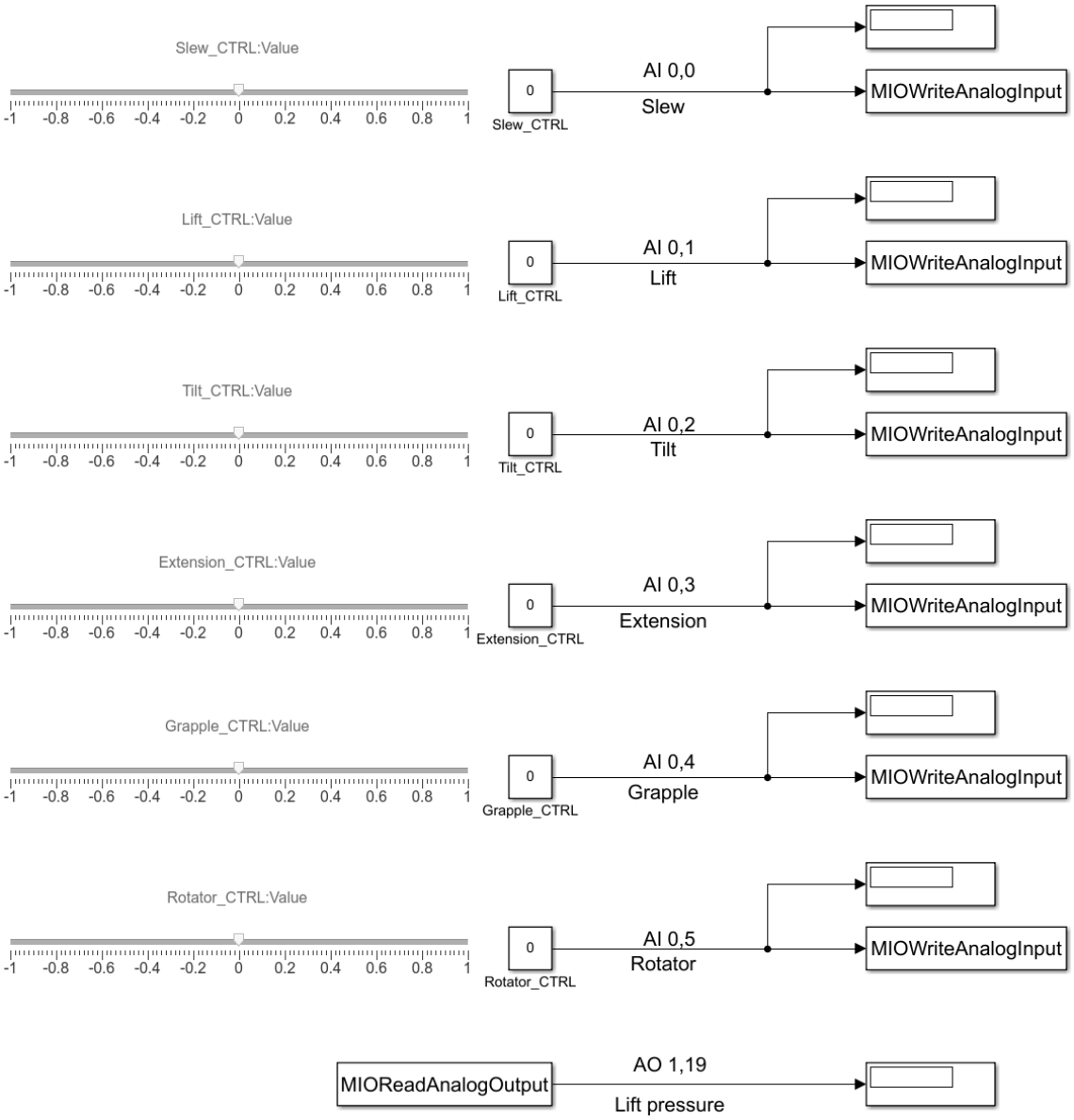

Figure 21. *Control system model in Simulink.*

The control signal that is sent to Amesim model's valves is controlled with sliders. The value of the sliders can be adjusted between -1 and 1. Both of the separated extension cylinders mentioned in section 4.2 are controlled by the same slider in the control system model, so the extension function works as desired in the mechanical model, i.e. synchronously. There are display blocks for every signal in the control system model, so it is easy to notice if there are some kind of major problems. The lowest display block shows the pressure in the B-chamber of the lift cylinder, which has been added to the model to test the data transfer from Amesim to Simulink. The "MIOWriteAnalogInput" and "MIOReadAnalogOutput" blocks are presented later in section 4.4.1.

4.4 All simulation models combined

At this point, all models presented in sections 4.1 - 4.3 were combined into a system simulation model. In other words, the whole simulation model now follows the basics of co-simulation presented in section 2.2.2, because Mevea does not yet support model exchange. Mevea is the co-simulation platform i.e. master software while Simulink and Amesim are slaves. The architecture of co-simulation in this model is presented in figure 22.

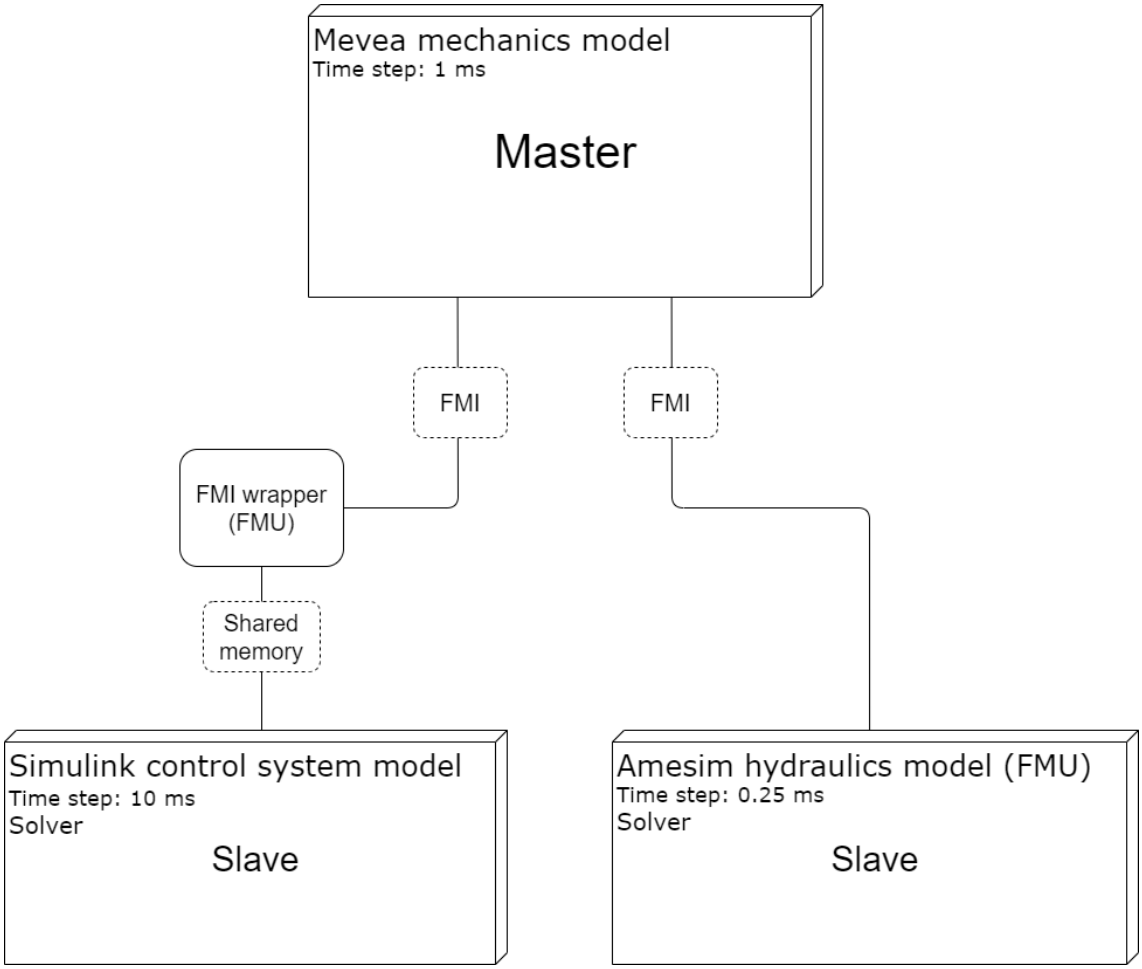

Figure 22. *Co-simulation architecture.*

As shown in figure 22, data is transferred between Mevea and Simulink and between Mevea and Amesim differently. There are two different interfaces (dashed line boxes): FMI and shared memory. These interfaces are described more specifically in the following sections 4.4.1 and 4.4.2. Simulink and Amesim both have their own solvers, which is specific to co-simulation. The co-simulation between Mevea and Amesim is implemented with generated code (see figure 5) and it is synchronous, but instead the co-simulation between Mevea and Simulink is implemented with tool coupling (see figure 6) and it is asynchronous because both software synchronize themselves.

4.4.1 Interfaces between Mevea and Simulink

The interfaces between Mevea and Simulink are a shared memory area and an FMI. Simulink uses Mevea's application programming interface functions that are located in MeveaIO.dll library. This method requires that Mevea I/O pool and Simulink are on the same PC. [26] There were given pre-made Simulink blocks made by Mevea Ltd, which were used to send and receive signals in Simulink. There were four different kind of blocks: send and receive blocks for analog signals and send and receive blocks for digital signals. In the model of this thesis, only blocks of analog signals were utilized (see figure 21). The Simulink blocks were made with C language, so they had to be compiled before use. The compiled blocks can be thought of as an FMI wrapper (see figure 22). The FMI wrapper is an FMU, which forms the FMI. In this thesis, the blocks were compiled with Visual Studio 2017.

There is a 10 ms time step in Simulink control system model, because if the time step is too small, it may cause the memory area to become clogged. The parameters that have to be set in Simulink are block number and channel number, on the same principle as in Mevea (see figure 18).

4.4.2 Interface between Mevea and Amesim

The interface between Mevea and Amesim is implemented by using functional mock-up unit (FMU) via functional mock-up interface (FMI). The FMI is created in Amesim and then the FMU is exported to Mevea. The FMU contains XML file and functions in C language. The FMI in Amesim is presented in figure 23.

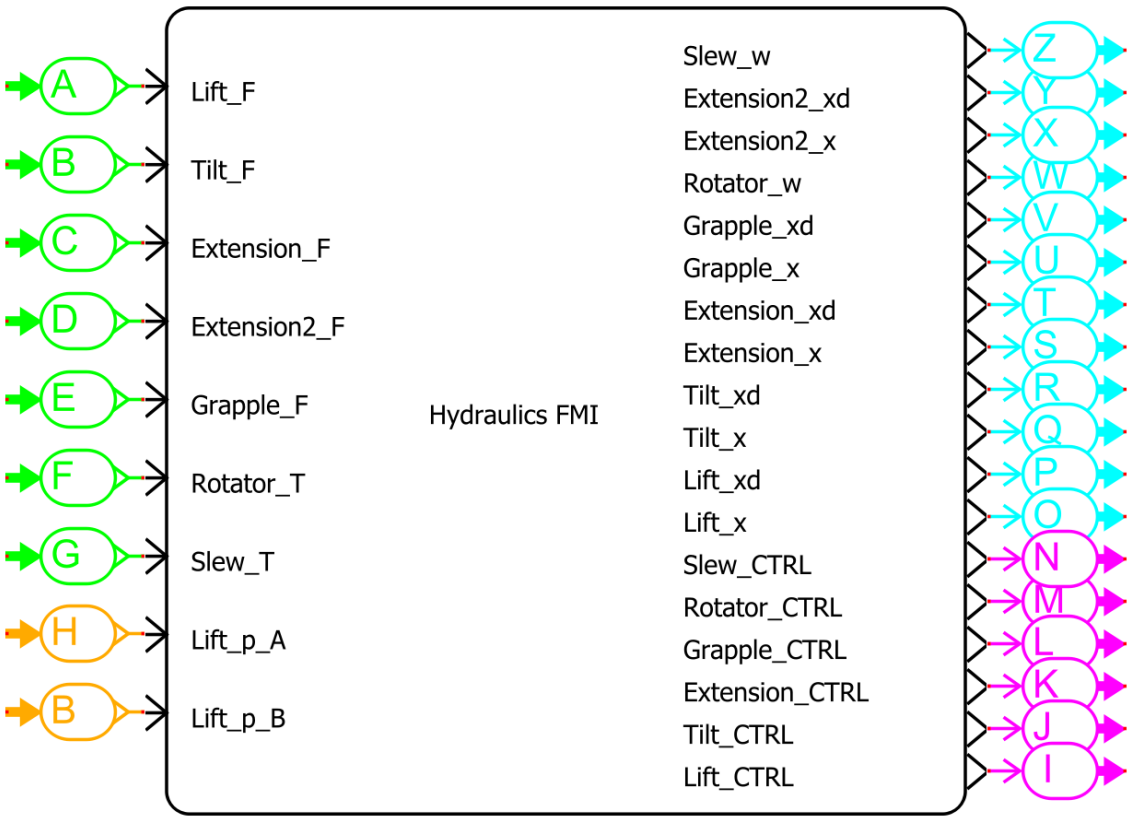

Figure 23. *Functional mock-up interface in Amesim.*

There are 9 inputs and 18 outputs in the FMI in figure 23. The inputs are cylinder forces (F), motor torques (T) and pressures (p) and the outputs are cylinder positions (x), cylinder velocities (xd), motor angular velocities (w) and control values (CTRL). These inputs of the FMI are inputs in Mevea, but they are outputs in Amesim. The same principle applies to the outputs of the FMI also. The FMI inputs and outputs are modelled with receivers (green and orange) and transmitters (blue and purple), which makes the visual appearance of the model neater. For example, receiver A in figure 23 (upper left corner) that connects to “Lift_F” input, gathers the signal from cylinder force transmitter A in figure 19. Respectively, the transmitter I at “Lift_CTRL” output in figure 23 (lower right corner) transmits the signal to the valve control signal receiver I in figure 19. The purple outputs are gathered from Simulink through Mevea and the blue inputs are gathered from Mevea. The green and orange inputs are sent to Mevea and they are calculated by Amesim model. The orange inputs, “Lift_p_A” and “Lift_p_B”, would not be needed in the model of this thesis at all, but they are made because the data transfer between Amesim and Simulink must be tested.

The hydraulic forces are calculated and updated on every time step, but it depends how many times. The more often forces are updates, the more accurate the model is, but then the model is also more cumbersome, which makes it more difficult to achieve real-

time simulation. Two different ways to solve FMU during one time step are presented in figure 24.

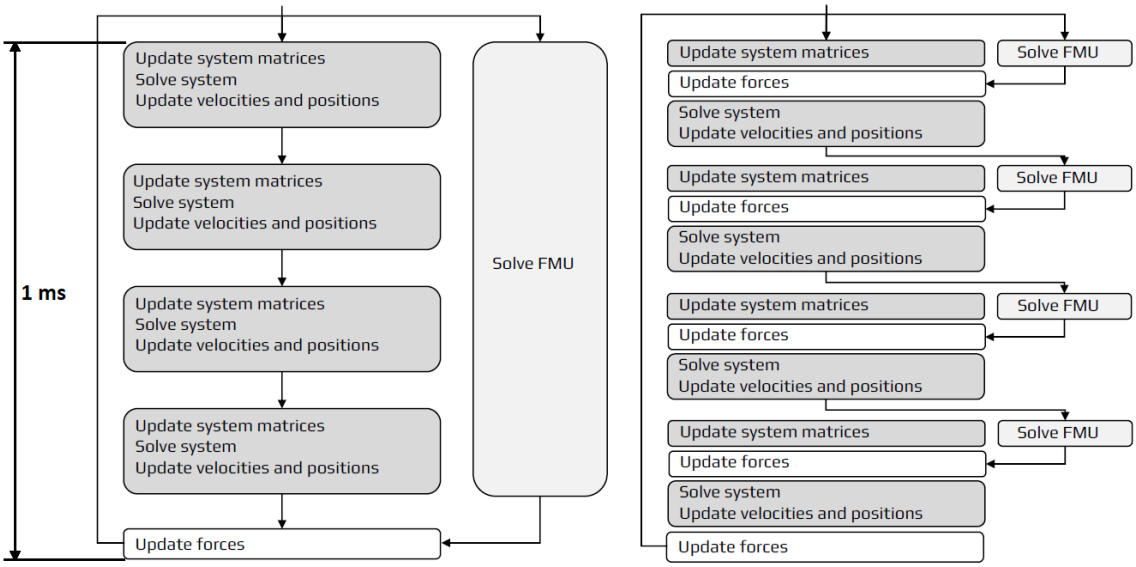

Figure 24. *Two different ways to solve FMU during one time step [27].*

In figure 24 there are two different ways of solving FMU during one time step: on the left side of the figure once in a time step and on the right side of the figure four times in a time step. In this model the time step in Mevea is 1 ms and the solution principle of the FMU is as shown in the right side of figure 24, because the numeric solver of the model is Runge-Kutta 4 (see figure 11). Thus, the time step in Amesim is a quarter of the time step in Mevea, i.e. 0.25 ms. Runge-Kutta 4 (fourth order Runge-Kutta) is a numerical integrating method of differential equations that calculates better approximations with better accuracy than first order Runge-Kutta that is shown on the left side of figure 24 [28].

After the FMI was exported to Mevea, the inputs and outputs had to be connected via user interface in Mevea. At this point, the logical naming of the signals simplifies and speeds up the work and reduces the risk of errors.

4.5 Simulating the system

This section describes how the system simulation was performed in practise. Simulation user interface in Mevea is presented in figure 25.

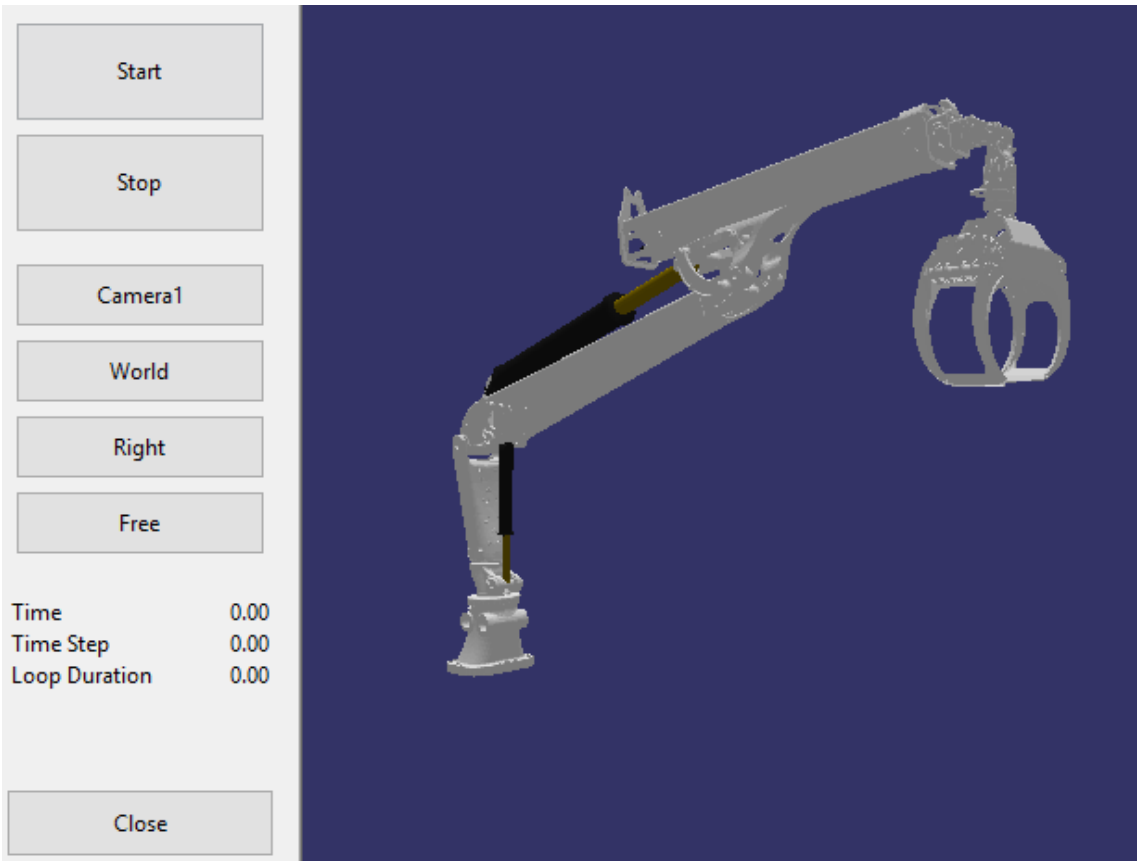

Figure 25. *Simulation user interface in Mevea.*

In Mevea, a dynamic simulation is started that shows the movements of the boom visually. At the same time, the Simulink model of the control system is run in the background to control the movements of the boom. Instead, Amesim does not need to be running at all, because the hydraulic model is included in the generated code of the exported FMU. When the simulation is started, the time step is displayed in seconds on the left. Below that is the time required to complete the computation in one time step (Loop Duration). If the loop duration is greater than the time step, the model cannot be run in real-time.

The real-time nature of the model of this thesis was tested so that as many different functions of the boom as possible were used simultaneously and in both directions. In other words, an attempt was made to make the computer's computation as cumbersome as possible. The results are presented in figure 26.

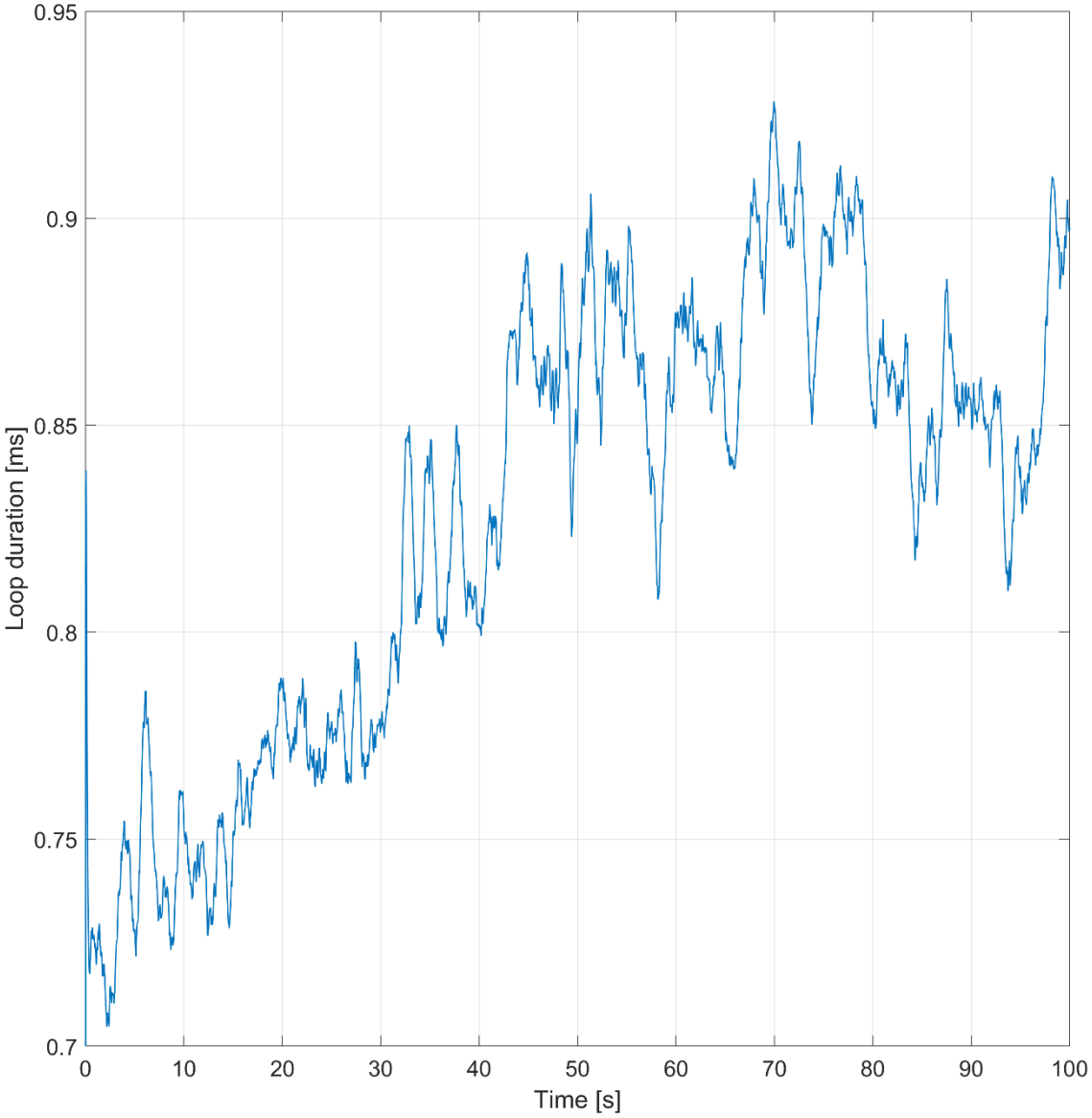

Figure 26. *Loop duration as a function of time.*

As it can be seen in figure 26, the simulation model is light enough to be simulated in real-time with 1 ms time step, as the computation performed during one time step takes less time than 1 ms. In this thesis, the model was simulated on a laptop computer with a 6-core Intel i7-8850H processor. According to Mevea user manual, a 2-core processor is a minimum requirement, because Mevea uses two different cores: one for simulation and one for user interface and visualization. In practise, the time step of the model could be reduced to about 0.93 ms, because the maximum value for time step in figure 26 is less than that at all times. On the other hand, the simulation lasted only 100 seconds, so it does not guarantee that the 0.93 ms calculation time will not be exceeded under any circumstances, so it is reasonable to keep the 1 ms time step.

A simple cycle in which the boom is raised and lowered at two different speeds is simulated and the results are presented in figure 27 and figure 28. The initial position of the boom is the same as presented in figure 25.

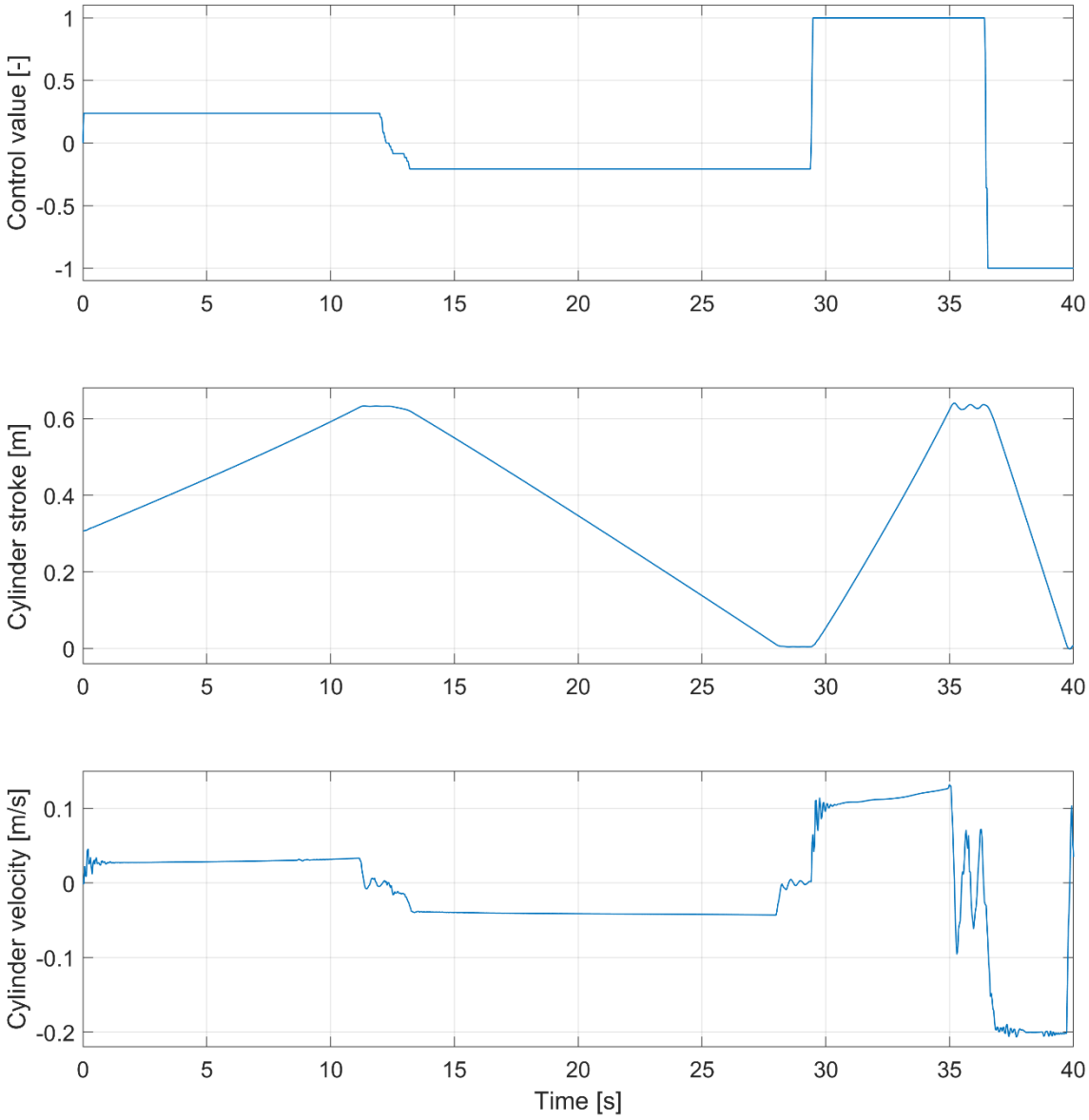

Figure 27. *Lift control value, lift cylinder stroke and lift cylinder velocity as a function of time.*

It can be seen in figure 27 that the lift cylinder is moving very smoothly at almost constant velocity while the control signal remains constant. The cylinder velocity is higher when lowering the boom, even though the control signal has the same absolute value as when raising the boom. This is explained by boom's own mass, gravity and the smaller surface area in B-chamber because of the cylinder rod. Slight oscillations can be observed when the cylinder is driven against its ends.

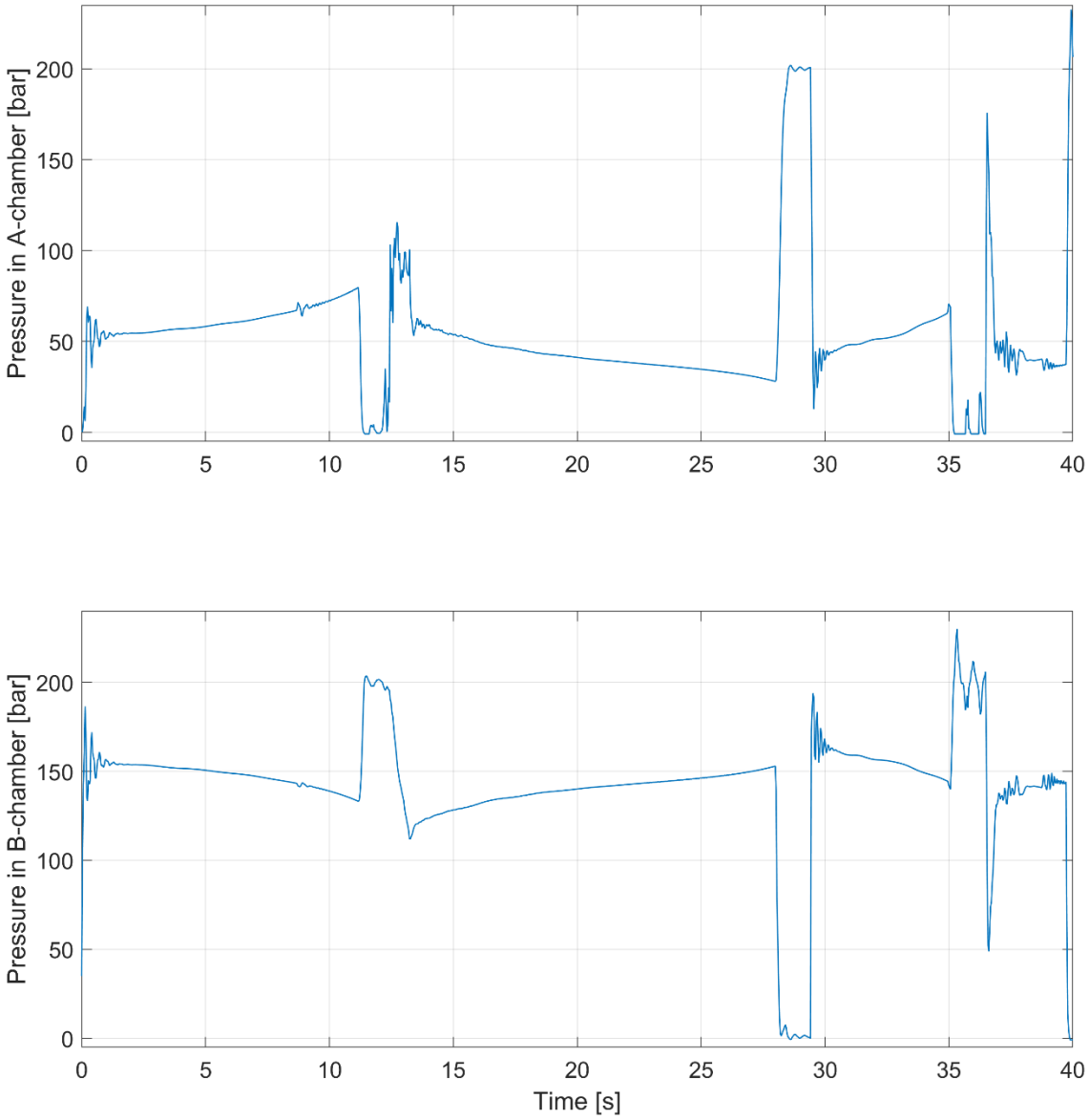

Figure 28. *Lift cylinder chamber pressures as a function of time.*

There are notable pressure peaks in figure 28 when cylinder reaches its ends. For instance, when the boom is raised as high as possible and the lift cylinder is driven to the end with the maximum stroke length, the pressure rise in B-chamber at lower velocity is 70 bar ($t \approx 12$ s) and at higher velocity 85 bar ($t \approx 35$ s). Instead, when lowering the boom as low as possible and the cylinder length is at its shortest, the pressure rise in A-chamber at lower velocity is 170 bar ($t \approx 27$ s) and at higher velocity 190 bar ($t \approx 39.5$ s). In general, the pressure in B-chamber is higher than in A-chamber, because the lift cylinder must support the boom continually.

4.6 Further development

Simulation models can be developed to be better and more accurate in practice indefinitely, but it is desirable to assess how much time is reasonable to spend on it. On this model of the thesis, it is important for further development to obtain more accurate and more realistic hydraulics and control system models in order to start verification of system simulation model with measured data. The suitability of other simulation tool combinations for system simulation could also be tested, for example modelling mechanics with Amesim or Simscape Multibody instead of Mevea. In Amesim there is a forwarder demo, based on which the modelling should be pretty straightforward. This demo has not been studied any further and it is downloaded directly from Amesim's demo library, but it is shown in figure 29 just as an example.

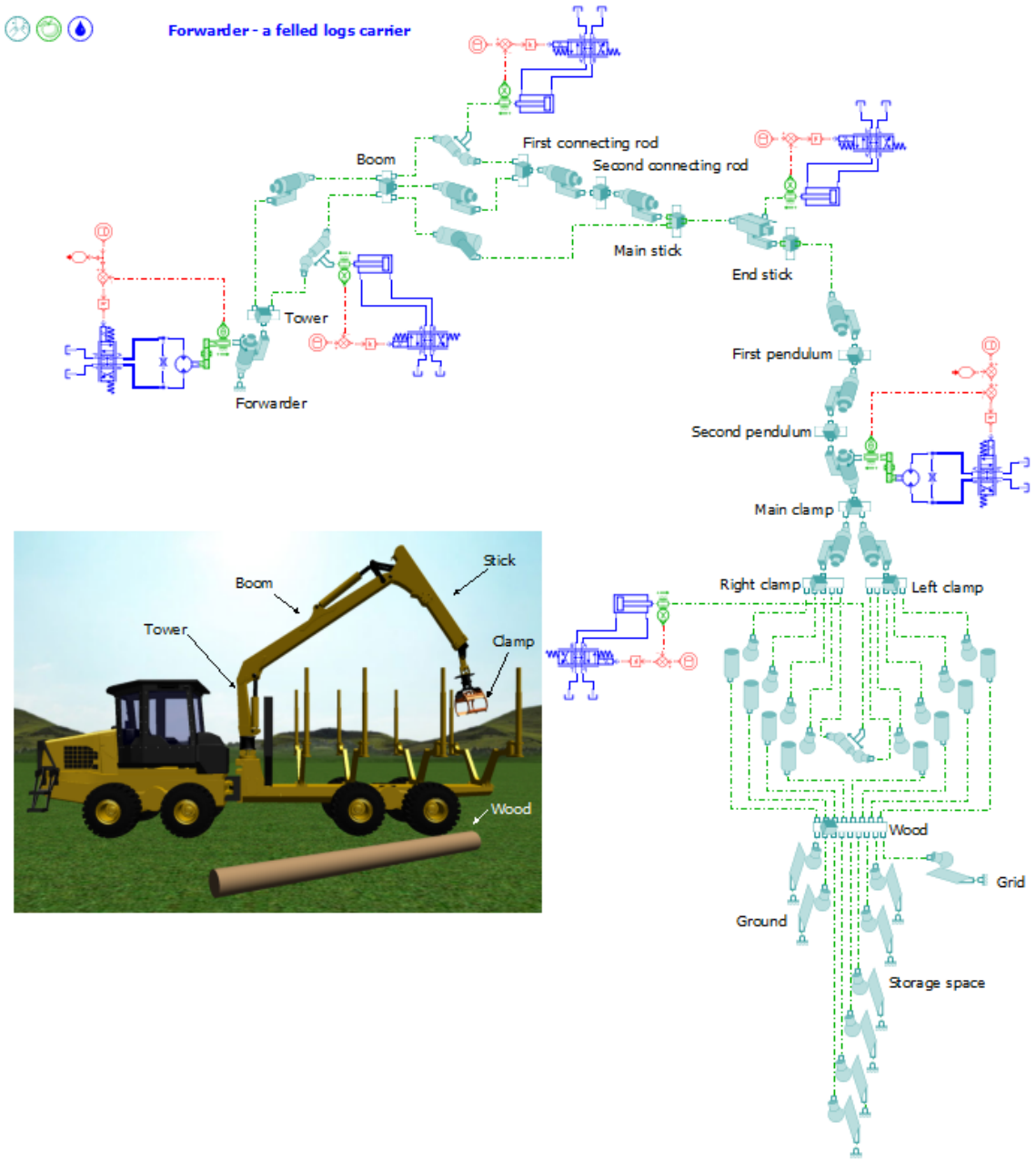

Figure 29. *Forwarder model demo in Amesim [29].*

The mechanical model in Mevea does not include collisions, so it could be the next step in the development process. In Mevea, collisions could be taken into account by creating collision graphics that must be really simple, so the computation does not become too cumbersome. Also, the boom control could be accomplished with a joystick instead of sliders in Simulink. There are also different modelling techniques that could be used to make the model more accurate. For example, the slewing of the boom could be modelled using rack and pinion approach instead of motor.

As the simulation models are made more realistic and more accurate, their computation also becomes more cumbersome. This problem could be solved by dividing the computation into many different PCs, for example by having three PCs running the mechanical, hydraulic and control system models each on their own.

5. CONCLUSIONS

The aim of this thesis was to study and map different simulation software for system simulation of Ponsse's forest machines. The survey included visits to various companies and meetings with software suppliers. The selected simulation software were Mevea (mechanics), Amesim (hydraulics) and Simulink (control system) and they were tested by making a simulation model of Ponsse's K121 loader. Mevea was the master software while Amesim and Simulink were slaves. The K121 simulation model built with the combination of simulation software mentioned above was able to run in real-time with a time step of 1 ms. Several different simulation areas, for example environment, electrics and power train, can be added to the simulation model while maintaining real-time simulation, because the model of this thesis was simulated on a laptop computer with a 6-core Intel i7-8850H processor instead of a powerful real-time PC.

The interfaces between the different software also worked as desired. The interface between Mevea and Amesim was implemented using FMU, which included XML file and standardized C functions. The FMU was generated from the Amesim's hydraulics simulation model and then exported to Mevea. The interface between Mevea and Simulink was implemented using an FMU wrapper and a shared memory area. Both software had access to the shared memory area through which they transmitted data to each other. Because Mevea was the master software, all data was transmitted through it, i.e. no data was transferred directly between Amesim and Simulink.

In the future it will be possible to simulate increasingly complex systems in real-time, as the computing power of computers seems to be constantly improving. According to Ponsse's future prospects, the next step is to test other simulation software as well, as the final suitability of the software for the company's needs will presumably only be noticed when using the software. In conclusion, the goals set for this thesis were achieved and the basis for system simulation was successfully created for the future.

REFERENCES

[1] N. Larrieu, A. Varet, Rapid Prototyping Software for Avionics Systems: Model-Oriented Approaches for Complex Systems Certification, 1st edition, John Wiley & Sons, Incorporated, Dec 3, 2014, 138 p.

[2] T. Higashi, Model-Based Design for Production Real-Time Embedded Systems, MATLAB and Simulink Consulting Services. Available (accessed Sep 10, 2020): https://se.mathworks.com/services/consulting/proven-solutions/model-based-design.html.

[3] JavaTpoint, V-Model. Available (accessed Sep 10, 2020) : https://www.javatpoint.com/software-engineering-v-model.

[4] Study.com, Why Scientists Use Models & Simulations, chapter 7, lesson 1. Available (accessed Aug 28, 2020): https://study.com/academy/lesson/why-scientists-use-models-simulations.html.

[5] E. Winsberg, Computer Simulations in Science, Stanford Encyclopedia of Philosophy, Sep 26, 2019. Available (accessed Aug 28, 2020): https://plato.stanford.edu/entries/simulations-science/.

[6] K. G. Shin, P. Ramanathan, Real-Time Computing: A New Discipline of Computer Science and Engineering, Proceedings of the IEEE, vol. 82, no. 1, Jan 1994. Available (accessed Sep 7, 2020): https://rtcl.eecs.umich.edu/papers/publications/1994/ramanathan-shin-ieee-proceedings.pdf.

[7] Siemens PLM Software, What is mechatronic system simulation?, presentation, Sep 29, 2016. Available (accessed Sep 8, 2020): https://www.slideshare.net/SiemensPLM/what-is-mechatronic-system-simulation.

[8] T. Farkas, A. Hinnerichs, C. Neumann, A Distributed Approach for Multi-Domain Simulation of Mechatronic Systems, 12th IEEE International Workshop on Future Trends of Distributed Computing Systems, 2008, Germany. Available (accessed Sep 9, 2020): https://ieeexplore.ieee.org/document/4683134.

[9] Modelica Association, Functional Mock-up Interface 2.0.1, specification, October 2, 2019, pp. 8-96. Available (accessed Aug 11, 2020): https://fmi-standard.org/.

[10] A. Vidanapathirana, S. D. Dewasurendra, S. G. Abeyaratne, Model in the loop testing of Complex Reactive Systems, IEEE 8th International Conference on Industrial and Information Systems, 2013, Sri Lanka. Available (accessed Sep 3, 2020): https://ieeexplore.ieee.org/document/6731950.

[11] T. Erkkinen, M. Conrad, Verification, Validation, and Test with Model-Based Design, 2008. Available (accessed Sep 3, 2020): https://www.mathworks.com/company/newsletters/articles/verification-validation-and-test-with-model-based-design.html.

[12] MathWorks support team, What is MIL, SIL, PIL, HIL and how do they integrate in Model Based Design approach?, MATLAB Answers, Jan 2, 2019. Available (accessed Sep 3, 2020): https://www.mathworks.com/matlabcentral/answers/440277-what-is-mil-sil-pil-hil-and-how-do-they-integrate-in-model-based-design-approach.

[13] Mevea Ltd, The Digital Twin at the center of R&D, p. 8.

[14] S. Haag, R. Anderl, Digital twin – Proof of concept, Manufacturing Letters, vol. 15, part B, Jan 2018, pp. 64-66. Available (accessed Sep 16, 2020): https://www.sciencedirect.com/science/article/abs/pii/S2213846318300208.

[15] P. Aivaliotis, K. Georgoulias, G. Chryssolouris, The use of Digital Twin for predictive maintenance in manufacturing, International Journal of Computer Integrated Manufacturing, vol. 32, Nov 13, 2019, pp. 1067-1080. Available (accessed Sep 16, 2020): https://www.tandfonline.com/doi/abs/10.1080/0951192X.2019.1686173?journalCode=tcim20.

[16] Ponsse Plc, The cut-to-length method. Available (accessed Feb 20, 2020): https://www.ponsse.com/products/cut-to-length#/.

[17] Ponsse Plc, About Ponsse. Available (accessed Feb 13, 2020): https://www.ponsse.com/en/web/guest/company/ponsse#/.

[18] Ponsse Plc, Ponsselta uutta voimaa kuormatraktorin kuormankäsittelyyn, 2018. Available (accessed Feb 13, 2020): https://www.ammattilehti.fi/uutiset.html?a2700=128858.

[19] ANSYS, Inc., Ansys Motion, 2020. Available (accessed Aug 24, 2020): https://www.ansys.com/products/structures/ansys-motion.

[20] ANSYS, Inc., Ansys Twin Builder, 2020. Available (accessed Aug 24, 2020): https://www.ansys.com/products/systems/ansys-twin-builder.

[21] MSC Software, Multibody Dynamics for Functional Virtual Prototyping, Adams brochure, Mar 2017. Available (accessed Aug 25, 2020): https://www.mscsoftware.com/product/adams.

[22] MathWorks, Simscape Multibody, documentation, 2020. Available (accessed Sep 1, 2020): https://www.mathworks.com/help/physmod/sm/.

[23] Siemens Industry Software Inc., Simcenter Amesim, 2020. Available (accessed Sep 17, 2020): https://www.plm.automation.siemens.com/global/en/products/simcenter/simcenter-amesim.html.

[24] Mevea Ltd, Mevea Modeller User manual, version 2.1.0.

[25] Mevea Ltd, Mevea Modeller: Beginner tutorials, version 2.0.

[26] A. Rouvinen, DSc (Tech), Senior Technical Advisor, Mevea Ltd, email, Aug 12, 2020.

[27] A. Rouvinen, DSc (Tech), Senior Technical Advisor, Mevea Ltd, Mevea co-simulation via FMI, training material.

[28] E. Cheever, Fourth Order Runge-Kutta, Svarthmore College study material. Available (accessed Jul 28, 2020): https://lpsa.swarthmore.edu/NumInt/NumIntFourth.html.

[29] Siemens Industry Software NV, Forwarder, demo simulation model, 2019.